HITACHI
Inspire the Next

uCosminexus Application Server

HTTP Server User Guide

3020-3-Y18(E)

■ Relevant program products

Applicable OSs: Windows Server 2008 x86, Windows Server 2008 x64[#1], Windows Server 2008 R2[#1]

P-2443-7K97 uCosminexus Application Server 09-00

P-2443-7S97 uCosminexus Service Platform 09-00[#2]

Applicable OSs: Windows Vista, Windows XP, Windows 7 (32-bit), Windows 7 (x64)[#1]

P-2443-7F97 uCosminexus Developer 09-00

P-2443-7T97 uCosminexus Service Architect 09-00[#2]

Applicable OSs: Windows Server 2008 x86, Windows Server 2008 x64[#1], Windows Server 2008 R2[#1], Windows Vista, Windows XP, Windows 7 (32-bit), Windows 7 (x64)[#1]

P-2443-7H97 uCosminexus Client 09-00

Applicable OS: Windows Server 2008 x64, Windows Server 2008 R2

P-2943-7K97 uCosminexus Application Server 09-00

P-2943-7S97 uCosminexus Service Platform 09-00[#2]

Applicable OS: AIX V6.1, AIX V7.1

P-1M43-7K92 uCosminexus Application Server (64) 09-00[#2]

P-1M43-7S92 uCosminexus Service Platform (64) 09-00[#2]

Applicable OS: HP-UX 11i V2 (IPF) , HP-UX 11i V3 (IPF)

P-1J43-7K92 uCosminexus Application Server 09-00

P-1J43-7S92 uCosminexus Service Platform 09-00[#2]

Applicable OS: Red Hat Enterprise Linux 5 (AMD/Intel 64) , Red Hat Enterprise Linux 6 (AMD/Intel 64)

P-9W43-7K92 uCosminexus Application Server 09-00[#2]

P-9W43-7S92 uCosminexus Service Platform 09-00[#2]

#1: Available only in WOW64 (Windows On Windows 64) environments.

#2: Confirm the support period for this product.

This manual can be used for the products other than products shown above. For details, see the *Release Notes*.

■ Export restrictions

If you export this product, please check all restrictions (for example, Japan's Foreign Exchange and Foreign Trade Law, and USA export control laws and regulations), and carry out all required procedures.

If you require more information or clarification, please contact your Hitachi sales representative.

■ Trademarks

■ Use of screenshots of Microsoft products

■ Restrictions

■ Issued

Aug. 2012: 3020-3-Y18(E)

■ Copyright

Preface

This manual provides an overview of Cosminexus Application Server and the BPM/ESB base. This manual explains the product components of Application Server and the BPM/ESB base, and the functionality of these products. It also explains the organization of the manuals for Application Server and the BPM/ESB base.

For Application Server and the BPM/ESB base, the following program products are used to build and operate the systems and to develop applications:

- uCosminexus Application Server
- uCosminexus Client
- uCosminexus Developer
- uCosminexus Service Architect
- uCosminexus Service Platform

This chapter describes the terms and abbreviations commonly used in the manuals for these program products.

Note that "Cosminexus" and "uCosminexus" might be abbreviated in this document.

Abbreviations for Application Server and BPM/ESB base products names

The following table shows the abbreviations used in this document for Application Server product names and the BPM/ESB base.

Abbreviation	Application Server and BPM/ESB base product name
Application Server	uCosminexus Application Server
Client	uCosminexus Client
Developer	uCosminexus Developer
Service Architect	uCosminexus Service Architect
Service Platform	uCosminexus Service Platform

Abbreviations for Application Server and BPM/ESB base function names

The following table shows the abbreviations used in this document for Application

Server function names and the BPM/ESB base.

Abbreviation	Application Server and BPM/ESB base function name
CJMSP broker	Cosminexus JMS provider broker function
CJMSP resource adapter	Cosminexus JMS provider resource adapter
Cosminexus Developer's Kit for Java	Cosminexus Developer's Kit for Java^{TM}
Cosminexus RM, or Reliable Messaging	Cosminexus Reliable Messaging
CSC	Cosminexus Service Coordinator
CTM	Cosminexus Component Transaction Monitor
DB Connector for Cosminexus RM	DB Connector for Cosminexus Reliable Messaging
HCSC	Hitachi Cosminexus Service Coordinator
HCSC-Business Process, Business Process base, or BP base	Hitachi Cosminexus Service Coordinator - Business Process
HCSC-Data Transform, or data conversion base	Hitachi Cosminexus Service Coordinator - Data Transform
HCSC-DB Adapter or DB adapter	Hitachi Cosminexus Service Coordinator - Database Adapter
HCSC-Manager or HCSC-MNG	Hitachi Cosminexus Service Coordinator - Manager
HCSC-Messaging, HCSC-MSG, or messaging base	Hitachi Cosminexus Service Coordinator - Messaging
HCSCTE	Hitachi Cosminexus Service Coordinator Tools for Eclipse
Management Server	Cosminexus Management Server
Manager	Cosminexus Manager
PRF	Cosminexus Performance Tracer
Smart Composer	Cosminexus Smart Composer
TPBroker	Cosminexus TPBroker

Abbreviations for related manuals

The following table shows the titles of related manuals, and related manuals used in this manual.

Application Server and BPM/ESB base related manuals

Manual title	Document number
uCosminexus Application Server First Step Guide	3020-3-Y00(E)
uCosminexus Application Server & BPM/ESB Platform Overview	3020-3-Y01(E)
uCosminexus Application Server System Setup and Operation Guide	3020-3-Y02(E)
uCosminexus Application Server Virtual System Setup and Operation Guide	3020-3-Y03(E)
uCosminexus Application Server System Design Guide	3020-3-Y04(E)
uCosminexus Application Server Web Container Functionality Guide	3020-3-Y05(E)
uCosminexus Application Server EJB Container Functionality Guide	3020-3-Y06(E)
uCosminexus Application Server Common Container Functionality Guide	3020-3-Y07(E)
uCosminexus Application Server Expansion Guide	3020-3-Y08(E)
uCosminexus Application Server Security Management Guide	3020-3-Y09(E)
uCosminexus Application Server Operation, Monitoring, and Linkage Guide	3020-3-Y10(E)
uCosminexus Application Server Maintenance and Migration Guide	3020-3-Y11(E)
uCosminexus Application Server Application Setup Guide	3020-3-Y13(E)
uCosminexus Application Server Command Reference Guide	3020-3-Y15(E)
uCosminexus Application Server Definition Reference Guide	3020-3-Y16(E)
uCosminexus Application Server Application and Resource Definition Reference Guide	3020-3-Y17(E)
uCosminexus Application Server HTTP Server User Guide	3020-3-Y18(E)
uCosminexus Application Server Reliable Messaging User Guide	3020-3-Y19(E)
uCosminexus Application Server Application Development Guide	3020-3-Y20(E)
uCosminexus Application Server API Reference Guide	3020-3-Y21(E)
uCosminexus Application Server XML Processor User Guide	3020-3-Y22(E)
uCosminexus Application Server Web Service Development Guide	3020-3-Y23(E)
uCosminexus Application Server SOAP Application Development Guide	3020-3-Y25(E)

Manual title	Document number
uCosminexus Application Server XML Security - Core User Guide	3020-3-Y26(E)
uCosminexus Application Server Messages	3020-3-Y27(E)

Generic names for a folder and a path

For content common among Windows, HP-UX, AIX, Linux, and Solaris, this manual uses the word *directory* to refer to a Windows *folder*, and uses a slash ("/") to refer to a backslash ("\"). For Windows, replace the word *directory* with *folder*, and "/" with "\".

Abbreviations for Microsoft products

The following table shows the abbreviations used in this document for Microsoft products.

Abbreviations		Product name
.NET Framework		Microsoft(R) .NET Framework Version 2.0
.NET Framework SDK		Microsoft(R) .NET Framework 2.0 Software Development Kit
Active Directory		Microsoft(R) Active Directory(R)
Windows Server Failover Cluster		Windows(R) Server Failover Cluster
Hyper-V		Microsoft(R) Hyper-V(TM)
Internet Explorer		Windows(R) Internet Explorer(R)
Internet Explorer 6		Microsoft(R) Internet Explorer(R) 6
Internet Explorer 7		Windows(R) Internet Explorer(R) 7
Internet Explorer 8		Windows(R) Internet Explorer(R) 8
Internet Explorer 9		Windows(R) Internet Explorer(R) 9
Microsoft IIS 7.0	Microsoft(R) Internet Information Services 7.0	Microsoft(R) Internet Information Services 7.0
Microsoft IIS 7.5	Microsoft(R) Internet Information Services 7.5	Microsoft(R) Internet Information Services 7.5
Excel		Microsoft(R) Office Excel
SQL Server 2005	SQL Server	Microsoft(R) SQL Server 2005
SQL Server 2008		Microsoft(R) SQL Server 2008

Abbreviations			Product name
SQL Server 2005 JDBC Driver	SQL Server JDBC Driver		Microsoft(R) SQL Server 2005 JDBC Driver
SQL Server JDBC Driver			Microsoft(R) SQL Server JDBC Driver 2.0
			Microsoft(R) SQL Server JDBC Driver 3.0
Microsoft Sysprep			Microsoft(R) Sysprep
Windows[#1]	Windows 7	Windows 7 x86	Microsoft(R) Windows(R) 7 Enterprise (32-bit)
			Microsoft(R) Windows(R) 7 Professional (32-bit)
			Microsoft(R) Windows(R) 7 Ultimate (32-bit)
		Windows 7 x64	Microsoft(R) Windows(R) 7 Enterprise (64-bit)
			Microsoft(R) Windows(R) 7 Professional (64-bit)
			Microsoft(R) Windows(R) 7 Ultimate (64-bit)
	Windows Server 2008	Windows Server 2008 x86	Microsoft(R) Windows Server(R) 2008 Enterprise 32-bit[#2]
			Microsoft(R) Windows Server(R) 2008 Standard 32-bit
		Windows Server 2008 x64	Microsoft(R) Windows Server(R) 2008 Enterprise[#2]
			Microsoft(R) Windows Server(R) 2008 Standard
		Windows Server 2008 R2	Microsoft(R) Windows Server(R) 2008 R2 Enterprise [#2]
			Microsoft(R) Windows Server(R) 2008 R2 Standard
	Windows Vista	Windows Vista Business	Microsoft(R) Windows Vista(R) Business
		Windows Vista Enterprise	Microsoft(R) Windows Vista(R) Enterprise
		Windows Vista Ultimate	Microsoft(R) Windows Vista(R) Ultimate

Abbreviations		Product name
	Windows XP	Microsoft(R) Windows(R) XP Professional Operating System
Windows Server Failover Cluster		Windows Server(R) Failover Cluster

Product names that include "x64" might be intended for Windows (x64), and product names that do not include "x64" might be intended for Windows (x86).

#1: "Windows (x86)" refers to Windows 32-bit. "Windows (x64)" refers to Windows 64-bit.

#2: "Windows Server 2008 Enterprise" might be used as a generic name to refer to this product.

Abbreviations for other products

The following table shows the abbreviations used in this document for other products than those from Microsoft.

Abbreviations		Product name
ACOS	AX2000	AX2000
	AX2500	AX2500
	BS320	BS320 Load Balancer Blade
AMD-V		AMD Virtualization
Application Server		uCosminexus Application Server
BIG-IP	BIG-IP v9	BIG-IP software version 9.1.0 or later
	BIG-IP v10.1	BIG-IP software version 10.1.0 or later
	BIG-IP v10.2	BIG-IP software version 10.2.0 or later
BJEX		Batch Job Execution System - Base uCosminexus Batch Job Execution Server
DCCM3		VOS1 DCCM3
		VOS3 XDM/DCCM3
Developer		uCosminexus Developer
Eclipse		Eclipse 3.6.1
HiRDB or HiRDB server	HiRDB Embedded Server, or embedded database	HiRDB Embedded Server Version 8
	HiRDB Server	HiRDB Server Version 9

Abbreviations		Product name
	HiRDB Server with Additional Function	HiRDB Server with Additional Function Version 9
	HiRDB/Parallel Server	HiRDB/Parallel Server Version 7
		HiRDB/Parallel Server Version 8
	HiRDB/Single Server	HiRDB/Single Server Version 7
		HiRDB/Single Server Version 8
HiRDB Run Time or HiRDB Client		HiRDB/Run Time Version 7
		HiRDB/Run Time Version 8
		HiRDB/Run Time Version 9
		HiRDB/Developer's Kit Version 8
		HiRDB/Developer's Kit Version 9
		HiRDB Developer's Suite Version 9
Intel VT		Intel(R) Virtualization Technology
IPF		Itanium(R) Processor Family
JP1		JP1 related products
JP1/AJS	JP1/AJS - Agent	JP1/Automatic Job Management System 2 - Agent#
		JP1/Automatic Job Management System 3 - Agent
	JP1/AJS - Manager	JP1/Automatic Job Management System 2 - Manager#
		JP1/Automatic Job Management System 3 - Manager
	JP1/AJS - View	JP1/Automatic Job Management System 2 - View#
		JP1/Automatic Job Management System 3 - View
JP1/AJS2	JP1/AJS2 - Agent	JP1/Automatic Job Management System 2 - Agent
	JP1/AJS2 - Light Edition	JP1/Automatic Job Management System 2 - Light Edition
	JP1/AJS2 - Manager	JP1/Automatic Job Management System 2 - Manager
	JP1/AJS2 - View	JP1/Automatic Job Management System 2 - View

Abbreviations			Product name
JP1/AJS3 or JP1/AJS	JP1/AJS3 - Agent		JP1/Automatic Job Management System 3 - Agent
	JP1/AJS3 - Manager		JP1/Automatic Job Management System 3 - Manager
	JP1/AJS3 - View		JP1/Automatic Job Management System 3 - View
JP1/AJS2 for Web Service			JP1/Automatic Job Management System 2 for Web Service
JP1/Cm2	JP1/Cm2/ESA		JP1/Cm2/Extensible SNMP Agent
JP1/ESP			JP1/Extensible Service Probe
JP1/File Transmission Server/FTP			JP1 Version 9 JP1/File Transmission Server/FTP
JP1/IM	JP1/IM - Manager		JP1/Integrated Management - Manager
	JP1/IM - View		JP1/Integrated Management - View
JP1/ITRM	JP1/ITRM - Manager		JP1/IT Resource Management - Manager
JP1/NETM/Audit			JP1/NETM/Audit - Manager
JP1/PFM			JP1/Performance Management
JP1/PFM	JP1/PFM - Agent	JP1/PFM - Agent for Cosminexus	JP1/Performance Management - Agent Option for uCosminexus Application Server
		JP1/PFM - Agent for Virtual Machine	JP1/Performance Management - Agent Option for Virtual Machine
	JP1/PFM - Base		JP1/Performance Management - Base
	JP1/PFM - Manager		JP1/Performance Management - Manager
	JP1/PFM - Web Console		JP1/Performance Management - Web Console
JP1/SC/DPM			JP1/ServerConductor/Deployment Manager Standard Edition
Loadflowbal			HA8000-ie/Loadflowbal
Oracle,	Oracle 11g		ORACLE$^{(R)}$ Oracle Database 11*g*$^{(TM)}$
or ORACLE			ORACLE$^{(R)}$ Oracle Database 11*g* R2$^{(TM)}$

Abbreviations			Product name
Process Modeler			Process Modeler 5 for Microsoft(R) Visio Professional Edition
SAP R/3			SAP R/3(R)
Service Architect			uCosminexus Service Architect
Service Platform			uCosminexus Service Platform
Struts			Jakarta Struts 1.1
TMS-4V/SP			Transaction Management System-4V/System Product
TMS-4V/SP/Server			Transaction Management System-4V/System Product/ Server
TP1/Base			uCosminexus TP1/Server Base
TP1/Client	TP1/Client/J		uCosminexus TP1/Client/J
	TP1/Client/P		uCosminexus TP1/Client/P
	TP1/Client/W		uCosminexus TP1/Client/W
TP1/COBOL adapter			TP1/COBOL adapter for Cosminexus
TP1/EE			TP1/Server Base Enterprise Option
TP1/Web			uCosminexus TP1/Web
UNIX	AIX		AIX V6.1
			AIX V7.1
	HP-UX or HP-UX (IPF)		HP-UX 11i V2 (IPF)
			HP-UX 11i V3 (IPF)
	Linux	Linux (IPF)	Red Hat Enterprise Linux(R) 5 Advanced Platform (Intel Itanium)
			Red Hat Enterprise Linux(R) 5 (Intel Itanium)
		Linux (x86/ AMD64 & Intel EM64T)	Red Hat Enterprise Linux(R) 5 Advanced Platform (AMD/Intel 64)
			Red Hat Enterprise Linux(R) 5 (AMD/Intel 64)
			Red Hat Enterprise Linux(R) 6 Advanced Platform (x86)

Abbreviations			Product name
			Red Hat Enterprise Linux(R) 6 (AMD/Intel 64)
			Red Hat Enterprise Linux(R) 6 (x86)
			Red Hat Enterprise Linux(R) 6 Advanced Platform (AMD/Intel 64)
			Red Hat Enterprise Linux(R) Server 6 (32-bit x86)
			Red Hat Enterprise Linux(R) Server 6 (64-bit x86_64)
	Solaris		Solaris 10
			Solaris 10 (SPARC)
			Solaris 10 (x64)
			Solaris 9
			Solaris 9 (SPARC)
VMware	VMware ESX		VMware(R) ESX
	VMware Tools		VMware(R) Tools
	VMware vCenter Server		VMware(R) vCenter Server
	VMware vSphere Client		VMware(R) vSphere Client
XDM/RD E2			Extensible Data Manager/Relational Database Extended Version 2

#: JP1/AJS2 might be used as a generic name.

As a generic name, application server is used for both Application Server and Developer.

For each version of Linux, the following abbreviations might be used.

Abbreviation	OS
Red Hat Enterprise Linux 5	Red Hat Enterprise Linux(R) 5 Advanced Platform (Intel Itanium)
	Red Hat Enterprise Linux(R) 5 (Intel Itanium)
	Red Hat Enterprise Linux(R) 5 Advanced Platform (AMD/Intel 64)
	Red Hat Enterprise Linux(R) 5 (AMD/Intel 64)
Red Hat Enterprise Linux Server 6	Red Hat Enterprise Linux(R) Server 6 (32-bit x86)
	Red Hat Enterprise Linux(R) Server 6 (64-bit x86_64)

Java related abbreviations

Abbreviation	Java term
BMP	Bean-Managed Persistence
BMT	Bean-Managed Transaction
BOM	Byte Order Mark
Connector 1.0	J2EE^{TM} Connector Architecture 1.0
Connector 1.5	J2EE^{TM} Connector Architecture 1.5
DI	Dependency Injection
EAR	Enterprise Archive
EJB or Enterprise JavaBeans	Enterprise JavaBeans^{TM}
EJB QL	EJB^{TM} Query Language
J2EE or Java 2 Platform, Enterprise Edition	J2EE^{TM}
	Java^{TM} 2 Platform, Enterprise Edition
J2SE	Java^{TM} 2 Platform, Standard Edition
JAAS	Java^{TM} Authentication and Authorization Service
JAR	Java^{TM} Archive
Java	Java^{TM}
Java 2 Runtime Environment, Standard Edition	Java^{TM} 2 Runtime Environment, Standard Edition
Java 2 SDK, or Java 2 SDK, Standard Edition	Java^{TM} 2 Software Development Kit, Standard Edition
JavaAPI	Java^{TM} Application Programming Interface
JavaBeans	JavaBeans^{TM}
Java EE, or Java Platform, Enterprise Edition	Java^{TM} Platform, Enterprise Edition
JavaMail	JavaMail^{TM}
Java SE	Java^{TM} Platform, Standard Edition

Abbreviation	Java term
JavaVM or JVM	Java^{TM} Virtual Machine
JAX-WS	Java^{TM} API for XML-Based Web Services
JAXB	Java^{TM} Architecture for XML Binding
JAXB or The Java Architecture for XML Binding	The Java^{TM} Architecture for XML Binding
JAXP or Java API for XML Processing	Java^{TM} API for XML Processing
JCA	J2EE^{TM} Connector Architecture
JCE	Java^{TM} Cryptography Extension
JDBC	Java^{TM} Database Connectivity
	JDBC^{TM}
JDK	Java^{TM} Development Kit
	JDK^{TM}
JMS	Java^{TM} Message Service
JMX	Java^{TM} Management Extensions
JNDI	Java Naming and Directory Interface^{TM}
JNI	Java^{TM} Native Interface
JPA	Java^{TM} Persistence API
JPQL	Java^{TM} Persistence Query Language
JSF	JavaServer^{TM} Faces Reference Implementation (RI) Version: 1.1_01 FCS
JSP	JavaServer Pages^{TM}
	JSP^{TM}
JSTL	JavaServer Pages^{TM} Standard Tag Library
JTA	Java^{TM} Transaction API
JTS	Java^{TM} Transaction Service

Abbreviation	Java term
POJO	Plain Old Java^{TM} Object
SAAJ	SOAP with Attachments API for Java^{TM}
Servlet or servlet	Java^{TM} Servlet
SMAP	Source Map
WAR	Web Archive

Abbreviations for general terms

The following table shows abbreviations used in this document for general terms.

Abbreviation	Term
ACL	Access Control List
ACOS	Advanced Core Operating System
AES	Advanced Encryption Standard
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
BLOB	Binary Large Object
BP	Business Process
BPEL	Business Process Execution Language
BPM	Business Process Management
BPMN	Business Process Modeling Notation
C14N	Canonicalization
CA	Certificate Authority
CMP	Container-Managed Persistence
CMR	Container-Managed Relationship
CMT	Container-Managed Transaction
CoC	Convention over Configuration
CORBA	Common Object Request Broker Architecture
CPU	Central Processing Unit

Abbreviation	Term
CR	Carriage Return
CRL	Certificate Revocation List
CSR	Certificate Signing Request
CSS	Cascading Style Sheets
CSV	Comma Separated Value
CUI	Character User Interface
CVS	Concurrent Versions System
DAO	Data Access Object
DB	Database
DBMS	Database Management System
DD	Deployment Descriptor
DDL	Data Definition Language
DES	Data Encryption Standard
DII	Dynamic Invocation Interface
DIT	Directory Information Tree
DMZ	Demilitarized Zone
DN	Distinguished Name
DNS	Domain Name System
DOM	Document Object Model
DoS	Denial of Service
DSA	Digital Signature Algorithm
DTD	Document Type Definition
DTO	Data Transfer Object
EIS	Enterprise Information System
EL	Expression Language
EOD	Ease of Development
ERP	Enterprise Resource Planning

Abbreviation	Term
ESB	Enterprise Service Bus
ETL	Extract Transform Loading
EUC	Extended UNIX Code
FAQ	Frequently Asked Questions
FF	Form Feed
FIFO	First-In First-Out
FK	Foreign Key
FLOPS	Floating point number Operations Per Second
FQDN	Fully Qualified Domain Name
FTP	File Transfer Protocol
GC	Garbage Collection
GIF	Graphic Interchange Format
GMT	Greenwich Mean Time
GPKI	Government Public Key Infrastructure
GUI	Graphical User Interface
HA	High Availability
HMAC	Hash based MAC
HNTRLib	Hitachi Network Objectplaza Trace Library
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Security
I/O	Input/Output
IANA	Internet Assigned Numbers Authority
ID	Identifier
IDE	Integrated Development Environment
IIOP	Internet Inter-Orb Protocol
IIS	Internet Information Services

Abbreviation	Term
IP	Internet Protocol
IPF	Itanium(R) Processor Family
ISAPI	Internet Server Application Programming Interface
iSCSI	Internet Small Computer System Interface
ISO	International Organization for Standardization
IT	Information Technology
IV	Initialization Vector
JIS	Japanese Industrial Standards
JSSE	Java Secure Socket Extension
JST	Japan Standard Time
JVMPI	Java Virtual Machine Profiler Interface
JVMTI	Java Virtual Machine Tool Interface
LAN	Local Area Network
LB	Load Balancer
LDAP	Lightweight Directory Access Protocol
LDIF	LDAP Data Interchange Format
LF	Line Feed
LGPKI	Local Government Public Key Infrastructure
MAC	Message Authentication Code
MB/s	Megabyte per Second
MBean	Managed Bean
Mbit/s	Megabit per Second
MDA	Model Driven Architecture
MDB	Message-Driven Bean
MHP	Message Handling Program
MIB	Management Information Base
MIME	Multipurpose Internet Mail Extensions

Abbreviation	Term
MIPS	Million Instructions Per Second
MTU	Maximum Transmission Unit
MVC	Model View Controller
NIC	Network Interface Card
NTP	Network Time Protocol
OAEP	Optimal Asymmetric Encryption Padding
OASIS	Organization for the Advancement of Structured Information Standards
OID	Object Identifier
OLTP	On-Line Transaction Processing
OMG	Object Management Group
ORB	Object Request Broker
OS	Operating System
OTM	Object Transaction Monitor
OTS	Object Transaction Service
PIM	Platform Independent Model
PK	Primary Key
PKI	Public Key Infrastructure
POA	Portable Object Adapter
POP3	Post Office Protocol - Version 3
PSM	Platform Specific Model
PTP	Point-to-Point
QName	Qualified Name
QoS	Quality of Service
RAC	Real Application Clusters
RAR	Roshal Archive
RD orRDB	Relational Database

Abbreviation	Term
REST	Representational State Transfer
RFC	Request For Comments
RMD	Reliable Messaging Destination
RMI	Remote Method Invocation
RMS	Reliable Messaging Source
RPC	Remote Procedure Call
RSA	Rivest, Shamir and Adleman
SaaS	Software as a Service
SAN	Storage Area Network
SAS	Serial Attached SCSI
SAX	Simple API for XML
SAX1	Simple API for XML 1.0
SAX2	Simple API for XML 2.0
SDK	Software Development Kit
SEI	Service Endpoint Interface
SFO	Session Fail Over
SHA	Secure Hash Algorithm
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SpecInt	Standard Performance Evaluation Corporation Integer benchmark
SPI	Service Provider Interface
SPP	Service Providing Program
ssh	Secure Shell
SSL	Secure Sockets Layer
SUP	Service Using Program

Abbreviation	Term
TCP	Transmission Control Protocol
TCS	Transaction Context Server
TLD	Tag Library Descriptor
TLS	Transport Layer Security
TrAX	Transformation API for XML
TSC	Time Stamp Counter
TSV	Tab Separated Values
UAC	User Account Control
UAP	User Application Program
UCS	Universal multi-octet coded Character Set
UDDI	Universal Description, Discovery and Integration
UML	Unified Modeling Language
UNC	Universal Naming Convention
UOC	User Own Coding
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
UTC	Universal Time Coordinated
UTF	UCS Transformation Format
UTF-8	8-bit UCS Transformation Format
VM	Virtual Machine
W3C	World Wide Web Consortium
WFA	Work-Flow Architecture
WS	Web Service
WSDL	Web Services Description Language
WSDL4J	Web Services Description Language for Java Toolkit
WS-I	Web Services Interoperability

Abbreviation	Term
WS-R	Web Services Reliability
WST	Web Standard Tools
WTP	Eclipse Web Tools Platform
XML	Extensible Markup Language
XPath	XML Path Language
XSL	Extensible Stylesheet Language
XSLT	XSL Transformations

Conventions: KB, MB, GB, and TB

This manual uses the following conventions:

- 1 KB (kilobyte) is 1,024 bytes.
- 1 MB (megabyte) is $1{,}024^2$ bytes.
- 1 GB (gigabyte) is $1{,}024^3$ bytes.
- 1 TB (terabyte) is $1{,}024^4$ bytes.

Contents

Chapter

1. Cosminexus HTTP Server

This chapter gives an overview of Cosminexus HTTP Server.

1.1 Overview of HTTP Server

Cosminexus HTTP Server is a web server for mission-critical systems used in mission-critical environments such as main systems or business systems, including those that perform transaction processing. Cosminexus HTTP Server supports high-reliability systems by providing comprehensive maintenance and technical services.

1.2 Features of Cosminexus HTTP Server

Cosminexus HTTP Server is developed based on Apache HTTP Server, which owns a large share of the global server market. This manual describes the features supported by Cosminexus HTTP Server.

The main features of Cosminexus HTTP Server are as follows:

- User authentication and access security
- Virtual host
- Reverse proxy
- Flow-restricting functionality
- Valid period settings functionality
- Header customization functionality
- Executing CGI programs
- Static contents cache functionality
- Displaying directory index
- Image map

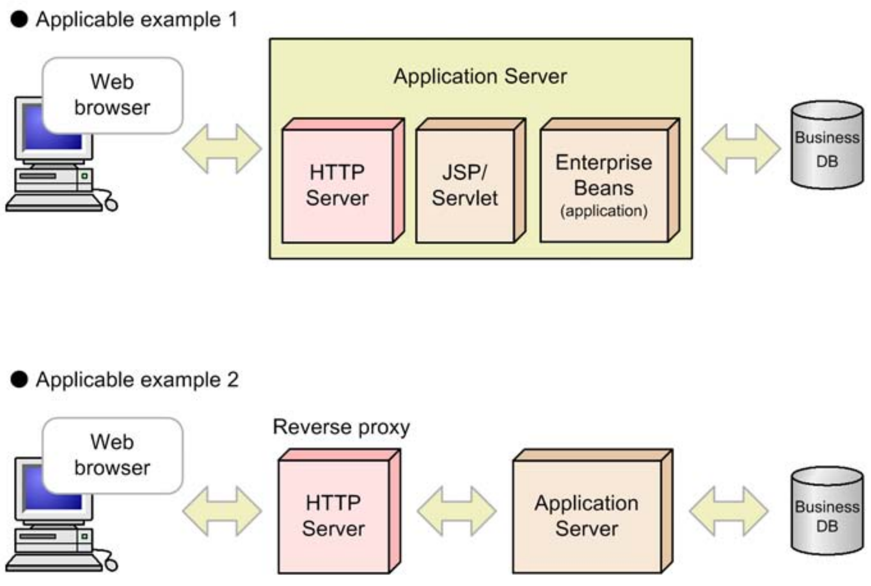
Additionally, Cosminexus HTTP Server uses BSAFE(R) SSL-C, a product of EMC Corporation, USA, and implements Secure Sockets Layer (SSL). Therefore, Cosminexus HTTP Server can prevent data tampering, spoofing (spoofing of the server from the client perspective, and spoofing of the client from the server perspective), and tapping, and ensures the security of information.

Example of applying uCosminexus Application Server

Cosminexus HTTP Server is one of the products that makes up uCosminexus Application Server.

The following shows an example of using Cosminexus HTTP Server.

Figure 1-1: Example of using Cosminexus HTTP Server



Chapter

2. Preparing, Starting, and Stopping Cosminexus HTTP Server (UNIX Version)

This chapter describes the points that you need to know before operating Cosminexus HTTP Server and also explains how to start and stop Cosminexus HTTP Server.

2.1 System configuration to operate Cosminexus HTTP Server

The section describes the system configuration required to operate Cosminexus HTTP Server.

(1) Hardware configuration

(a) Server

For details about supported models, memory requirements, and disk space requirements for Cosminexus HTTP Server, see the *Release Notes* document.

(b) Client

The required client is a terminal that can run a Web browser.

(c) Network

- Network such as Ethernet (mandatory)
- Domain name system server (optional)
- Load balancer (optional)
- SSL accelerator (optional)
- Firewall (optional)

2.2 Installing and uninstalling Cosminexus HTTP Server

Cosminexus HTTP Server can be used after installing uCosminexus Application Server.

The following are the procedures for installing and uninstalling uCosminexus Application Server. For details, see the *uCosminexus Application Server System Setup and Operation Guide*.

- To install

  Use the Hitachi Integrated Installer. Cosminexus HTTP Server will be installed in the following directory:

  /opt/hitachi/httpsd

- To uninstall

  Use the Hitachi PP Installer.

2.3 Defining the operating environment

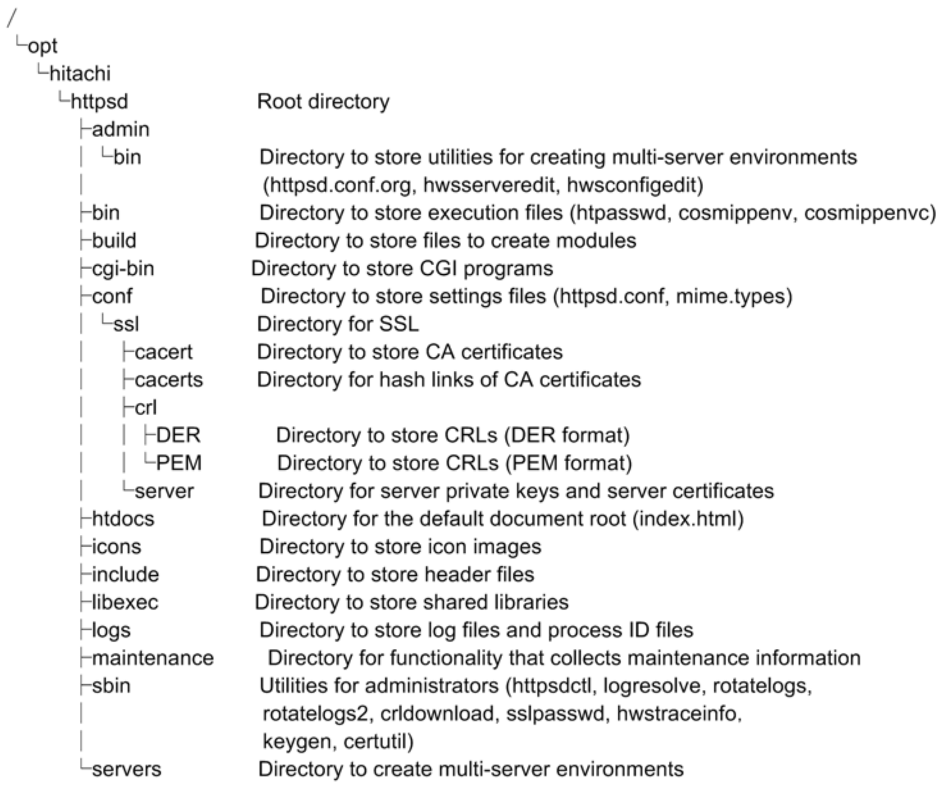
This section describes the file that defines Cosminexus HTTP Server operations.

2.3.1 How to define the environment

(1) Directory configuration

The following figure shows the directory configuration when Cosminexus HTTP Server is installed. Do not change this configuration:

Figure 2-1: Directory configuration



```
/
└opt
  └hitachi
    └httpsd                  Root directory
      ├admin
      │ └bin                 Directory to store utilities for creating multi-server environments
      │                       (httpsd.conf.org, hwsserveredit, hwsconfigedit)
      ├bin                   Directory to store execution files (htpasswd, cosmippenv, cosmippenvc)
      ├build                 Directory to store files to create modules
      ├cgi-bin               Directory to store CGI programs
      ├conf                  Directory to store settings files (httpsd.conf, mime.types)
      │ └ssl                 Directory for SSL
      │   ├cacert            Directory to store CA certificates
      │   ├cacerts           Directory for hash links of CA certificates
      │   ├crl
      │   │ ├DER               Directory to store CRLs (DER format)
      │   │ └PEM               Directory to store CRLs (PEM format)
      │   └server            Directory for server private keys and server certificates
      ├htdocs                Directory for the default document root (index.html)
      ├icons                 Directory to store icon images
      ├include               Directory to store header files
      ├libexec               Directory to store shared libraries
      ├logs                  Directory to store log files and process ID files
      ├maintenance             Directory for functionality that collects maintenance information
      ├sbin                  Utilities for administrators (httpsdctl, logresolve, rotatelogs,
      │                       rotatelogs2, crldownload, sslpasswd, hwstraceinfo,
      │                       keygen, certutil)
      └servers               Directory to create multi-server environments
```

(2) Configuration file

The file that defines the operating environment of Cosminexus HTTP Server is called a *configuration file*. You cannot specify full-width characters or Unicode supplementary characters outside of comment lines in the configuration file.

The following table describes the application of these files.

Table 2-1: Application of the configuration files

File name	Application	Standard Provision
httpsd.conf	This file defines the operating environment of Cosminexus HTTP Server with various directives. The System Administrator manages the file.	Y
mime.types	This file defines the relationship between the file extension of content and the content type (MIME type). The system administrator manages the file.	Y
.htaccess	The access control file defines access control. The end-user creates this file as required, under the directory for which access is to be controlled (The default file name is `.htaccess`).	N
Files specified in the Include directive	These files define the operating environment of Cosminexus HTTP Server by using directives. The system administrator manages the files. mod_jk.conf (redirector operating definition file for Cosminexus HTTP Server) is typically one of these files. For details on mod_jk.conf, see *9.3 mod_jk.conf (redirector operating definition file for Cosminexus HTTP Server)*in the manual *uCosminexus Application Server Definition Reference Guide*.	N

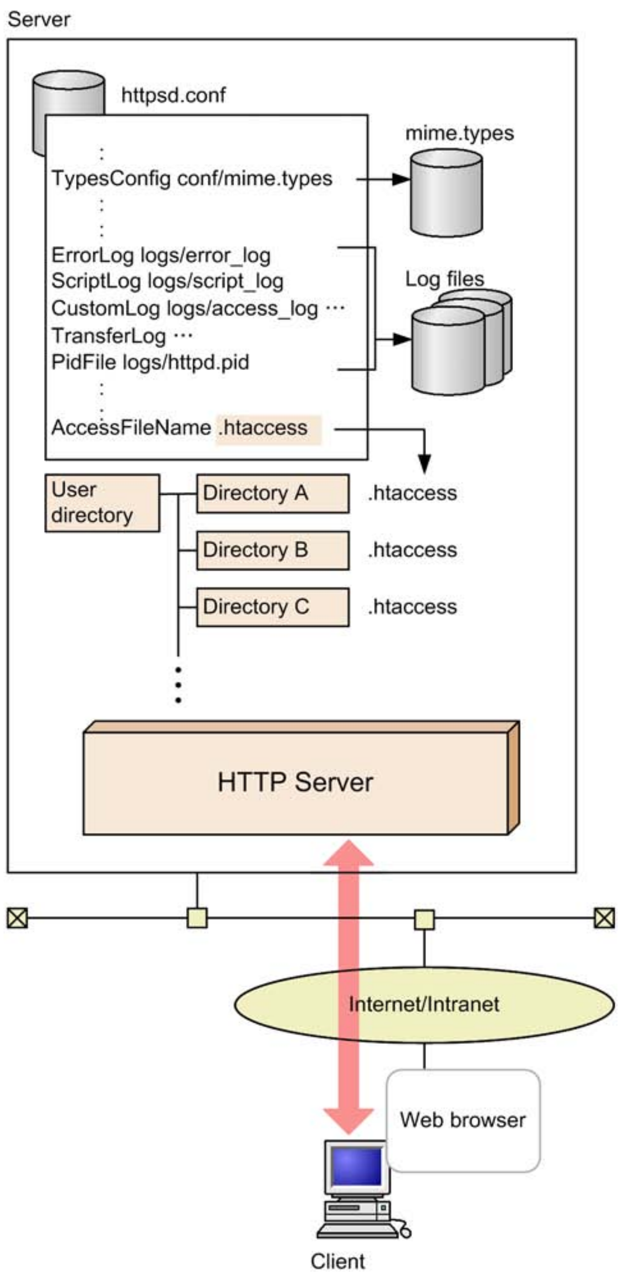
Legend:

Y: Provided.

N: Not provided.

The following figure shows the relationship between configuration files.

Figure 2-2: Relationship between configuration files



2.3.2 How to define system parameters

This section describes how to define system parameters required for Cosminexus HTTP Server. For details on the settings, see the manual of the OS that is used.

(1) Maximum size of the shared memory segment

In Cosminexus HTTP Server, the status information of the server process and the internal trace information are collected in the shared memory. The OS contains a system parameter that controls the maximum size of each segment of the shared memory.

The size (in bytes) of the shared memory that is used in the status information of the server process is 400 KB, and the trace information is `'the MaxClients directive value × 7 KB'`. In the system parameter of maximum size of the shared memory segment, set a value that is equal to or more than the above-mentioned value.

The following is an example of the system parameter. The type and contents of system parameter in the OS differ based on versions of the used OS or kernel. For details on the contents of system parameter and the setup method, see the manual of the OS that is used.

In Linux(R)

Maximum size of the shared memory segment: `kernel.shmmax` (`/etc/sysctl.conf`)

In AIX and HP-UX (IPF)

Maximum size of the shared memory segment: `shmmax`

(2) Maximum number of processes

The number of processes on the system and the number of processes for each user are controlled by an OS system parameter. Out of the following processes, consider the total number of start processes that exist in the operating environment, and then set the system parameter:

Control process

Number of start processes: 1

System parameters to be changed: Maximum number of processes for the user who starts the Web server.

Server process

Number of start processes: Value specified in the MaxClients directive

System parameters to be changed: Maximum number of processes for the user specified in the User directive.

CGI process

When the execution of the CGI program is allowed, the CGI process is invoked from each server process when the request is processed.

Number of start processes: The value specified in the MaxClients directive

Number of system parameters to be changed: The maximum number of processes for the user specified in the User directive.

gcache server

The *gcache* server starts when the SSL session management functionality is enabled.

Number of start processes: 1

System parameters to be changed: The maximum number of processes for the user specified in the User Directive.

rotatelogs process

Start *rotatelogs process* to divide log entries up by specific time periods.

Number of start processes: The number of rotatelogs programs specified in the CustomLog, ErrorLog, HWSRequestLog, and TransferLog directives.

System parameters to be changed: Maximum number of processes for the user who starts the Web server.

rotatelogs2 process

Start *rotatelogs2 process* to split the log entries up based on the log file size and output them to multiple log files that wrap around when they all get full.

Number of start processes: The number of rotatelogs2 programs specified in the CustomLog, ErrorLog, HWSRequestLog, and TransferLog directives.

System parameters to be changed: Maximum number of processes for the user

who starts the Web server.

For the process structure of Cosminexus HTTP Server, see the *4.1.1 Architecture of Cosminexus HTTP Server process (UNIX Version)*.

The following is an example of a system parameter. The type and contents of system parameter of the OS differ depending on the version of the OS or the kernel being used. For details on contents of system parameters and setup method, see the manual of the OS being used.

In Linux

Maximum number of processes for the entire system: `kernel.threads-max` (`/etc/sysctl.conf`)

Maximum number of processes per user: `nproc` (`/etc/security/`

`limits.conf`)

In AIX

Maximum number of processes per user: `maxuproc`

In HP-UX (IPF)

Maximum number of processes for the entire system: `nproc`

Maximum number of processes per user: `maxuprc`

(3) Maximum number of files (Number of file descriptors)

The number of files that can be opened on the system, and the number of files that can be opened by users are restricted depending upon the system parameters of the OS. Consider the following file descriptors to be used in Cosminexus HTTP Server, and then set the system parameters:

In Linux

Number of file descriptors =

$(50 + A \times B + C + 11 \times C \times D + 8 \times E + (F + I) \times G) \times 1.2$

In AIX

Number of file descriptors =

$(50 + A \times B + C + 3 \times C \times D + 5 + (F + H)) \times 1.2$

In HP-UX (IPF)

Number of file descriptors =

$(50 + A \times B + C + 3 \times C \times D + 5 \times E + (F + H) \times G) \times 1.2$

(Legend)

A: Number of specified Listen directives (When the Listen directive is not specified: 1)

B: Number of IP addresses allocated to the host

C: Total number of specified CustomLog, ErrorLog, HWSRequestLog, and TransferLog directives

D: When the rotatelogs or rotatelogs2 program is used: 1; When neither program is used: 0

E: Number of concurrently executing CGI scripts (value specified for MaxClients)

F: When SSL is used: 3; when SSL is not used: 2

G: Number of concurrently executing requests (value specified for MaxClients)

H: When reverse proxy is used: 1; when reverse proxy is not used: 0

Also note that the number of file descriptors used in CGI programs or used in external modules that are not a part of Cosminexus HTTP Server, are not included.

(4) Example of system parameter definitions

The following is an example of a system parameter. The type and contents of the system parameter in the OS differ according to the version of the OS or kernel being used. For details on the contents of system parameters and the setup method, see the manual of the OS being used.

In Linux

- Maximum number of files for the entire system: `fs.file-max` (`/etc/sysctl.conf`)

In AIX

In AIX, set the number of file descriptors that a process can open.

- Soft limit for the number of file descriptors that a user process can open at once: `nofiles`

  For nofiles, specify a value equal to or greater than the calculated estimate.
- Hard limit for the number of file descriptors that a user process can open at once: `nofiles_hard`

For nofiles_hard, specify a value equal to or greater than the value specified for nofiles.

In HP-UX (IPF)

- Maximum number of files for the entire system: `nfile`
- sInitial value for the logical maximum number of file descriptors per process: `maxfiles`

  For maxfiles, specify a value equal to or greater than the calculated estimate.
- Physical maximum number of file descriptors per process: `maxfiles_lim`

For maxfiles_lim, specify a value equal to or greater than the value specified for maxfiles.

2.4 Starting and stopping

This section explains how to start and stop Cosminexus HTTP Server.

2.4.1 Starting and stopping Cosminexus HTTP Server(Using Management Server)

You can use Management Server to start or stop Cosminexus HTTP Server. For details, see *4.1.2 How to start the system* or *4.1.4 How to stop the system* in the manual *Cosminexus Application Server Operations Management Portal Operation Guide*.

2.4.2 Starting and stopping Cosminexus HTTP Server (httpsdctl command)

This section describes the httpsdctl command that starts and stops Cosminexus HTTP Server.

(1) Format

```
/opt/hitachi/httpsd/sbin/httpsdctl {start | stop | restart | graceful |
gracefulstop | configtest | help}
```

(2) Options

- start

  This option starts Cosminexus HTTP Server. If you are using an encrypted private key for SSL, a request to enter the private key password is displayed when starting Cosminexus HTTP Server.

- stop

  This option stops Cosminexus HTTP Server.

- restart

  This option restarts Cosminexus HTTP Server (server restart). End the running server processes immediately. After all the server processes end, restart Cosminexus HTTP Server. When you restart Cosminexus HTTP Server, the change in the value specified in the MaxClients directive is not applied, and the value that is set earlier will continue to be used. If you changed the value specified in the Listen directive and the settings for the private key used in SSL communication (that is, the SSLCertificateKeyFile directive), stop and restart Cosminexus HTTP Server.

- graceful

  Restart Cosminexus HTTP Server. End the running server processes, and then stop the server. Start the server process with a new configuration file when required. When you restart Cosminexus HTTP Server, the change in the value that is specified in the MaxClients directive is not applied, and the value that is set earlier will continue to be used. If you changed the value specified in the Listen directive and the settings for the private key used in SSL communication (that is, the SSLCertificateKeyFile directive), stop and restart Cosminexus HTTP Server.

- gracefulstop

  This option stops Cosminexus HTTP Server. End the running server processes, and then stop the server. If you do not end the running processes, the processes will end when the waiting time specified in the HWSGracefulStopTimeout directive ends.

- configtest

  This option checks the configuration file syntax. If there is a syntax error, an error message is displayed on the screen. If you specify the `configtest` option, Cosminexus HTTP Server will not start.

- help

  This option displays the httpsdctl help.

(3) How to confirm the start

To confirm the start of Cosminexus HTTP Server, check the control process. For details, see *(3) Monitoring the control process* of *4.1.3 Operation management*.

(4) Example of usage

The following example starts Cosminexus HTTP Server. If you are using an encrypted private key, enter the password.

```
/opt/hitachi/httpsd/sbin/httpsdctl start
Enter PEM pass phrase:
```

(5) Notes

- When you stop the Web server with `httpsdctl stop` and `gracefulstop`, if the configuration file of Cosminexus HTTP Server is not correctly defined, an error will occur in the httpsdctl execution and the Web server will not stop.
- When you restart the Web server with `httpsdctl restart` and `graceful`, if the configuration file of Cosminexus HTTP Server is not defined correctly, an error will occur in the httpsdctl execution and the Web server will not restart.
- When you start, restart, and stop Cosminexus HTTP Server with the httpsdctl command, the message showing the completion of start and stop is not output.

2.4.3 Starting Cosminexus HTTP Server (httpsd command)

You can also use the httpsd command to start Cosminexus HTTP Server. Normally, you do not use this method. You use this method to specify an `httpsd.conf` file name other than the default name or to specify the root directory and the `httpsd.conf` file when starting the server.

If you are using an encrypted private key for SSL, a request to enter the private key password is displayed when starting Cosminexus HTTP Server.

(1) Format

```
/opt/hitachi/httpsd/sbin/httpsd [[-d directory] [-f file-name] [-R directory] | -v
| -t]
```

(2) Options

- -d *directory*

  You can specify the default value used when the ServerRoot directive is not specified in the configuration file.

- -f *file-name*

  You can specify the `httpsd.conf` file. Specify the `httpsd.conf` file with the absolute path or the relative path from the value specified in the ServerRoot directive.

- -R *directory*

  This option specifies the absolute path of the directory storing the DSO execution library.

- -v

  This option displays the version information. If you specify this option, Cosminexus HTTP Server will not start.

- -t

  This option checks the configuration file syntax. If there is a syntax error, an error message will be displayed on the screen. If you specify this option, Cosminexus HTTP Server will not start.

(3) How to restart

You can use the `kill` command to restart Cosminexus HTTP Server:

```
kill {-HUP | -USR1} `cat PidFile-directive-specified-value`
```

- -HUP

  Restart as when using the `restart` option of the httpsdctl command.
- -USR1

  Restart as when using the `graceful` option of the httpsdctl command.
- *PidFile-directive-specified-value*

  Specify a value (file name) that is specified in the PidFile directive.

(4) How to end

When you use the httpsd command to start Cosminexus HTTP Server, execute the following command to end the process and stop Cosminexus HTTP Server:

```
kill {-TERM | -USR2} `cat PidFile-directive-specified-value`
```

- -TERM

  Stop as when using the `stop` option of the httpsdctl command.
- -USR2

  Stops as when using the `gracefulstop` option of the httpsdctl command.

(5) How to confirm the start

To confirm the start of Cosminexus HTTP Server, check the control process. For details, see *(3) Monitoring the control process* of *4.1.3 Operation management*.

2.4.4 Operation by general user account

For Cosminexus HTTP Server, normal operation is assumed to be operation by the superuser.

When Cosminexus HTTP Server is installed, various settings are configured for operation by the superuser.

Thus, when users other than the superuser (hereafter referred to as *general users*) operate Cosminexus HTTP Server, they need to change the settings file for Cosminexus HTTP Server and settings in related directories and files. For some functionality in Cosminexus HTTP Server, some operations are restricted from general users.

This section describes the differences between the superuser and general users, and methods to create an environment for general users to operate Cosminexus HTTP Server, and the restrictions thereof.

(1) Permissions for each process

The following table lists the permissions of each process for operation by the superuser or general users.

Table 2-2: Permissions for each process

No.	Process	Operation by the superuser	Operation by general users
1	Control process	Superuser	General user
2	`rotatelogs` and `rotatelogs2` processes		
3	Server process	Users or groups specified in the User and Group directives	
4	CGI process		
5	`gcache` server		

(2) Differences between the superuser and general users in UNIX

In UNIX, unlike general users, the superuser has system administrator permissions. The following table lists examples of the differences between the superuser and general users in UNIX.

Table 2-3: Table Example differences between the superuser and general users in UNIX

No.	Item	Superuser	General user
1	Can stop processes that were started by users?	Yes	No
2	Can open well-known ports (ports 1023 and lower)?	Yes	No
3	Can access files that do not explicitly have read or write permissions?	Yes	No

If a general user operates Cosminexus HTTP Server, because the control process in Cosminexus HTTP Server operates with general user permission, the behavior in this case might differ from operation by the superuser. Therefore, if a general user operates Cosminexus HTTP Server, the user needs to create an environment while considering the differences with the superuser.

(3) Changing resource owners and groups

In UNIX, you can change resource owners and groups for content and settings files for Cosminexus HTTP Server, and for files and directories accessed by Cosminexus HTTP Server during operation.

At the minimum, you will need to change the resources under the installation directory (`/opt/hitachi/httpsd`).

If you want to restore resource owners and groups to the previous settings, save the owners and groups for the current resources before making changes.

The superuser can save owners and groups. The following is an example of how to do this.

Example:

For the resources under the `/opt/hitachi/httpsd` directory, create a list of owners and groups.

```
ls laR /opt/hitachi/httpsd
```

The superuser can change owners and groups. The following is an example of how to do this.

Example:

For the resources under the `/opt/hitachi/httpsd` directory, change the owner (`hwsuser`) and the group.

```
chown R hwsuser:hwsgroup /opt/hitachi/httpsd
```

(4) Starting httpsd

Use the general user who operates Cosminexus HTTP Server to start `httpsd`.

To stop or restart `httpsd`, use the general user who started `httpsd`.

(5) Restrictions

The commands below cannot be operated by general users. Operate these commands as the superuser.

- `crldownload`
- `htpasswd`
- `hwscollect`
- `hwsserveredit`
- `logresolve`
- `sslccert`
- `sslckey`

- `sslpasswd`

In operation by general users, the following directives cannot be specified. Any directive specified by general users is ignored.

- Group
- User

In operation by general users, well-known ports (ports 1023 and lower) cannot be opened.

Be careful when specifying the port number in the following directives:

- Listen
- Port
- SSLCacheServerPort

Chapter

3. Preparing, Starting, and Stopping (Windows Version)

This chapter describes the points that you need to know before operating Cosminexus HTTP Server and also explains how to start and stop Cosminexus HTTP Server.

3.1 System configuration to operate Cosminexus HTTP Server

This section describes the system configuration required to operate Cosminexus HTTP Server.

(1) Hardware configuration

(a) Server

For details about supported models, memory requirements, and disk space requirements for Cosminexus HTTP Server, see the *Release Notes* document.

(b) Client

The required client is a terminal that can run a Web browser.

(c) Network

- Network such as Ethernet (mandatory)
- Domain name system server (optional)
- Load balancer (optional)
- SSL accelerator (optional)
- Firewall (optional)

(2) Notes on using Cosminexus HTTP Server in Windows

(a) Notes on executing commands

To run Cosminexus HTTP Server on Windows 7, Windows Vista, Windows Server 2008 R2, or Windows Server 2008, you need to execute all commands in this manual with Administrator permissions. Execute the Cosminexus HTTP Server commands by using an elevated command prompt. (The window title for an elevated command prompt is Administrator: Command Prompt.) Open the Administrator Command Prompt by using the functionality provided in the OS.

(b) Notes on updating settings files

When you update settings files of Cosminexus HTTP Server in Windows 7, Windows Vista, Windows Server 2008 R2, or Windows Server 2008, be sure to execute the update program with Administrator permissions.

3.2 Installing and uninstalling

Cosminexus HTTP Server can be used by installing uCosminexus Application Server.

The following are the procedures for installing and uninstalling uCosminexus Application Server. For details, see the *uCosminexus Application Server System Setup and Operation Guide*.

- To install

  Use the Hitachi Integrated Installer. Cosminexus HTTP Server will be installed in the following directory:

  <*Cosminexus-installation-directory*>\httpsd

- To uninstall

  Use the Hitachi PP Installer.

3.3 Definition files of operating environment

This section describes configuration files that define Cosminexus HTTP Server operations.

(1) Directory configuration

The following figure shows the directory configuration when Cosminexus HTTP Server is installed. Do not change this configuration.

Figure 3-1: Directory configuration

```
Application Server installation directory
 └httpsd          Root directory (httpsd.exe)
   ├admin
   │ └bin          Directory for storing multiple server environment generation utilities
   │                 (httpsd.conf.org, hwsserveredit.exe, hwsconfigedit.exe)
   ├bin           Directory for storing executable files (htpasswd.exe)
   ├cgi-bin        Directory for storing CGI programs
   ├conf           Directory for configuration files (httpsd.conf, mime.types)
   │ └ssl          Directory for SSL
   │    ├cacert      Directory for storing CA certificates
   │    ├crl
   │    │ ├DER       Directory for storing CRLs (in DERs)
   │    │ └PEM       Directory for storing CRLs (in PEMs)
   │    └server      Directory for the server private keys and server certificates
   ├htdocs         Default document root directory (index.html)
   ├icons          Directory for storing icon images
   ├include        Directory for storing header files
   ├libexec        Directory for shared libraries
   ├libldap        Location in which the LDAP libraries are stored
   ├logs           Directory for storing logs and process ID files
   ├modules          Directory for storing modules (mod_hws_ldap.so, mod_proxy.so,
   │              mod_expires.so, mod_headers.so, mod_hws_cache.so,
   │              mod_hws_qos.so, mod_proxy_http.so)
   ├sbin           Utilities for administrators (logresolve.exe, rotatelogs.exe,
   │              rotatelogs2.exe, crldownload.exe, sslpasswd.exe, hwstraceinfo.exe,
   │              keygen.exe, certutil.exe)
   └servers        Directory for generating multi-server environments
```

(2) Configuration file

The file that defines the operating environment of Cosminexus HTTP Server is called a *configuration file*. You cannot specify full-width characters or Unicode supplementary characters outside of comment lines in the configuration file.

The following table describes the application of these files.

Table 3-1: Application of the configuration files

File name	Application	Standard Provision
httpsd.conf	This file defines the operating environment of Cosminexus HTTP Server with various directives. The System Administrator manages the file.	Y
mime.types	This file defines the relationship between the file extension of content and the content type (MIME type). The system administrator manages the file.	Y
.htaccess	The access control file defines access control. The end-user creates this file as required, under the directory for which access is to be controlled (The default file name is `.htaccess`).	N
Files specified in the Include directive	These files define the operating environment of Cosminexus HTTP Server by using directives. The system administrator manages the file. mod_jk.conf (redirector operating definition file for Cosminexus HTTP Server) is typically one of these files. For details on mod_jk.conf, see *9.3 mod_jk.conf (redirector operating definition file for Cosminexus HTTP Server)* in the manual *uCosminexus Application Server Definition Reference Guide.*	N

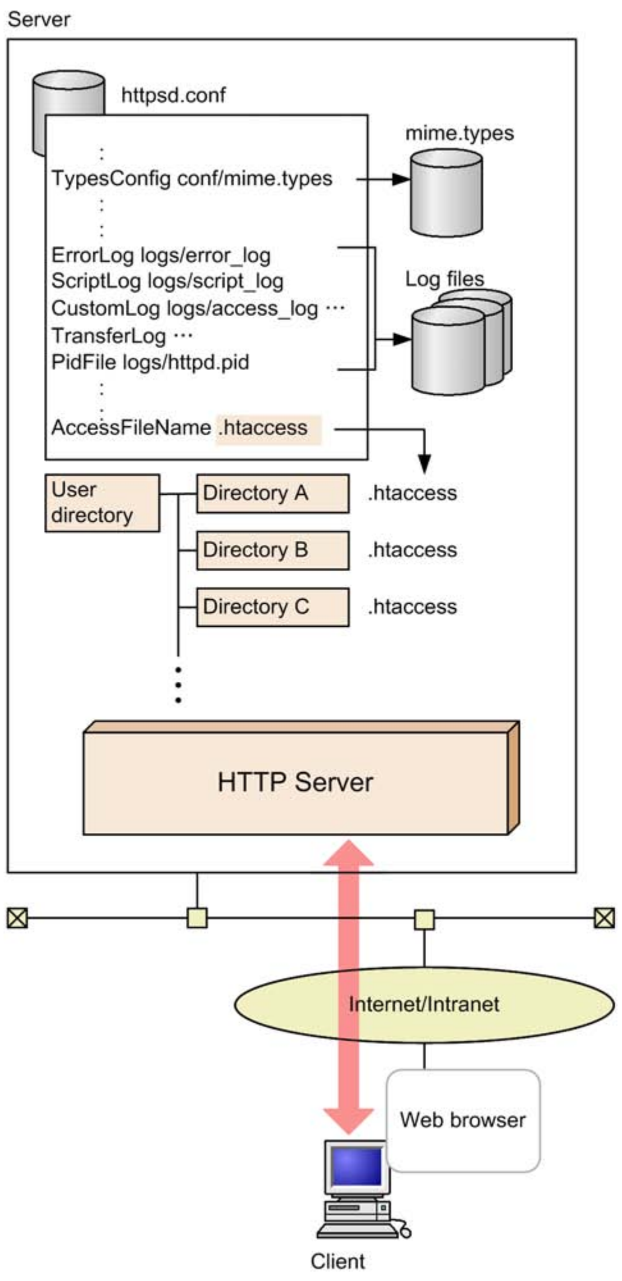
Legend:

Y: Provided.

N: Not provided.

The following figure shows the relationship between the configuration files.

Figure 3-2: Relationship between configuration files



3.4 Starting and stopping

This section explains how to start and stop Cosminexus HTTP Server.

3.4.1 Starting and stopping Cosminexus HTTP Server

When you install Cosminexus HTTP Server, it is registered in the system as a service called *Cosminexus HTTP Server*. The server is registered as a service to be started manually. the service does not start automatically when the OS is started.

To start, stop, and restart Cosminexus HTTP Server, you can use any of the following methods:

- Starting, stopping, and restarting the service from Management Server
- Starting and stopping as a service from the Control Panel
- Starting, stopping, and restarting from the Command Prompt

When executing Cosminexus HTTP Server as a service, the user account will be "LocalSystem" at the time of installation. Cosminexus HTTP Server including the CGI program and the API connection module is executed by this user account. For operation by other user accounts, see *3.4.2 Operation by general user accounts*.

(1) Starting, stopping, and restarting the service from Management Server

For details, see *4.1.2 How to start the system* or *4.1.4 How to stop the system* in the manual *Cosminexus Application Server Operations Management Portal Operation Guide*.

(2) Starting and stopping as a service from the control panel

Open the Service window from the Control Panel, select **Cosminexus HTTP Server**, and then click the **Start** button to start Cosminexus HTTP Server, or click the **Stop** button to stop Cosminexus HTTP Server. You cannot restart Cosminexus HTTP Server from the Service window.

(3) Starting, stopping, and restarting from the Command Prompt

Enter the `httpsd` command from the command prompt. The `httpsd` command is coded below:

(a) Format

```
"<Cosminexus-installation-directory>\httpsd\httpsd.exe" [[-d directory][-f file-name][
[-n "Service-name"][-k {start | stop | restart | gracefulstop | install |
uninstall} ] ]| -v | -t ]
```

(b) Options

- -d *directory*

  Use this option to set the default value used when the ServerRoot directive is not specified in the configuration file.

- -f *file-name*

  Use this option to specify the `httpsd.conf` file. Specify the file name with an absolute path, or with a relative path from the specified value of the ServerRoot directive.

- -n "*Service-name*"

  Use this option to specify the service name of Cosminexus HTTP Server. Specify the service name within " (quotation marks). You can specify a maximum of 128 characters in the service name. Specify the service name in ASCII code. You cannot specify the following characters in the service name:

  '￥', '/', '"', control codes, and multi-byte characters

  The default value of the service name is `Cosminexus HTTP Server`.

  If you specify this option, you also need to specify the `-k` option.

- -k start

  Use this option to start Cosminexus HTTP Server. When the `-n "Service name"` is specified, the corresponding service starts.

- -k stop

  Use this option to stop Cosminexus HTTP Server. When the `-n "Service name"` is specified, the corresponding service stops.

- -k restart

  Use this option to restart Cosminexus HTTP Server.

- -k gracefulstop

  Use this option to stop Cosminexus HTTP Server. The option stops the server after ending the running server threads. If running server threads do not end, the server threads will end when the waiting time specified in the HWSGracefulStopTimeout directive ends.

- -k install

  Use this option to register Cosminexus HTTP Server as a service. When the `-n "service name"` is specified, the corresponding service is registered. When registering the service, the startup type will be 'Manual'. When Cosminexus HTTP Server starts as a service, it sets the default value of the ServerRoot directive to the path specified in the `httpsd.exe` command or to the value specified in the `-d` option.

- -k uninstall

  Use this option to delete Cosminexus HTTP Server from the service. When the `-n "service name"` is specified, the corresponding service is deleted. If the service is running, first stops the service and then deletes the service.

- -v

  Use this option to display the version information. Cosminexus HTTP Server does not start when you specify this option.

- -t

  Use this option to check the configuration file syntax. If there is a syntax error, an error message is displayed on the screen. Cosminexus HTTP Server does not start when you specify this option.

(4) Operating Cosminexus HTTP Server from a remote machine using the terminal service

By using the terminal service functionality in Windows 7, Windows Vista, Windows XP Professional, Windows Server 2008 R2, or Windows Server 2008, you can start and stop Cosminexus HTTP Server, or execute commands on the server computer from a remote machine.

For details on operating a terminal service, see the OS manual.

(5) Notes

When Cosminexus HTTP Server is stopped from **Control Panel**, or by the `-k stop` option executed from a command prompt, if a server thread is running, it will stop after waiting up to 30 seconds.

3.4.2 Operation by general user accounts

When executing Cosminexus HTTP Server as a service, the user account is LocalSystem at the time of installation. Cosminexus HTTP Server, including CGI programs and the API connection module, is executed by this user account.

This section describes how to operate Cosminexus HTTP Server by using a general user account to which only permissions required for operation have been set, without belonging to a group that has various permissions.

(1) Creating a general user account

This section describes how to create a general user account to start Cosminexus HTTP Server service.

How to create a general user account

1. From the Control Panel, open **Administrative Tools**, and then **Computer Management**.
2. In **Computer Management**, open **System Tools**, **Local Users and Groups**, and then **Users**.
3. From the **Action** menu, select **New User**, and then enter the necessary information.

   Be sure to enter a password. Also, specify whether the password never expires.

By default, group settings are added to a created general user account. Execute the following procedure to delete the group settings.

How to delete group settings

1. From the Control Panel, open **Administrative Tools**, and then **Computer Management**.
2. In **Computer Management**, open **System Tools**, **Local Users and Groups**, and then **Users**.
3. Show the **Properties** of the new user, and then display the **Member Of** tab.
4. Delete the registered groups.

(2) Assigning the user permissions

This section describes how to assign user permissions to the created general user account.

How to assign user permissions

1. From the Control Panel, open **Administrative Tools**, and then **Local Security Policy**.
2. Open **Security Settings**, **Local Policies**, and then **User Rights Assignment**.
3. Double-click **Log on as a Service** to open it.
4. Click the **Add user or group** button, and then add the corresponding user account.

Even if you do not explicitly specify the **Log on as a Service** permission, the permission is automatically added to the general user that changed the service logon account. For details about changing the service logon account, see *(3) Changing the service logon account*.

(3) Changing the service logon account

This section describes how to change the Cosminexus HTTP Server service logon account to the general user account.

How to change the service logon account

1. From the Control Panel, open **Administrative Tools**, and then **Services**.
2. Display the **Properties** of the **Cosminexus HTTP Server** service, and then open the **Log On** tab.
3. Select the **This account** radio button, and then specify the general user account. Enter the password that you specified in *(1) Creating a general user account* correctly.

(4) Specifying access permissions for directories and files

Add full control permissions for the created general user account to the access permissions for directories and files that Cosminexus HTTP Server accesses.

(5) Starting the service

Start the Cosminexus HTTP Server service by using an account that permission to start services. The general user account does not this permission.

(6) Notes

To use the `hwstraceinfo` command, execute it from a general user account specified in *(3) Changing the service logon account*. You cannot execute the command by using a user account with Administrators permissions.

Chapter

4. How to Operate the System

This chapter explains how to set up a Web server environment using directives and commands according to operations.

4.1 Relationship between processes and directives of Cosminexus HTTP Server

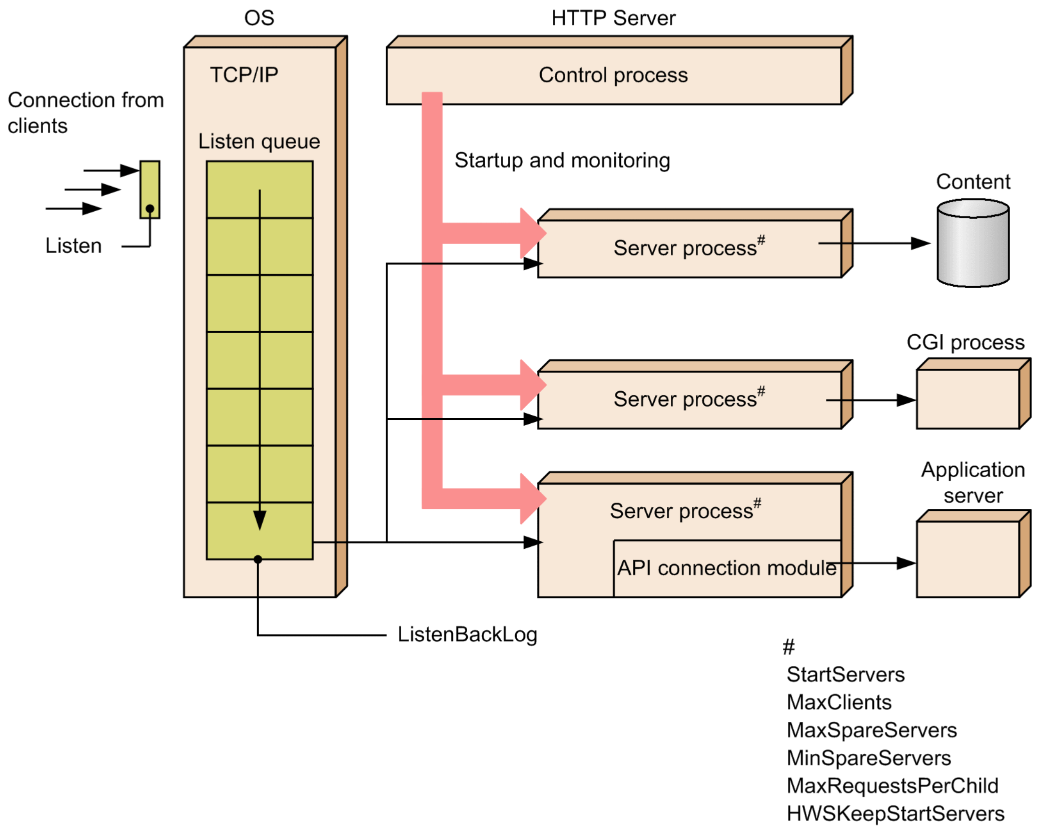
This section describes the relationship between processes and directives of Cosminexus HTTP Server.

4.1.1 Architecture of Cosminexus HTTP Server process (UNIX Version)

(1) Process architecture

The following figure shows the architecture of Cosminexus HTTP Server process.

Figure 4-1: Architecture of Cosminexus HTTP Server process (UNIX Version)



When you start Cosminexus HTTP Server, the control process starts. The control process starts the server process to process the request, and monitors the server process operations. Initially, the control process generates the number of server processes specified in the StartServers directive. Later, the control process increases or decreases the number of server processes based on values specified in the MinSpareServers and MaxSpareServers directives. The MaxClients directive specifies the maximum value for number of server processes. The control process manages the increase or decrease in the number of server processes. This process is called *maintenance*.

The OS receives the TCP connection of client, from the IP address and port specified in the Listen directive, and reserves the connection in the Listen queue of OS. You can specify the size of the Listen queue in the ListenBacklog directive. The TCP connection that cannot be stored in the Listen queue is not established. One of the server processes picks up a TCP connection stored in the Listen queue and performs the processing.

One server process accepts a single TCP connection and performs the processing. One server process ends after processing the number of HTTP requests specified in the MaxRequestsPerChild directive. In such a case, the control process generates a new server process and continues processing.

The control process is operated by user and group permissions with which Cosminexus HTTP Server is started. The server process is operated with user and group permissions specified in the User and Group directives. For both, control processes and server processes, the process name (execution program name) is `httpsd`. The process ID of the control process is output to the file specified in the PidFile directive.

(2) Transition of number of processes

To avoid the load concentration on the server, the maintenance generates 2^{n-1} server processes (n is the number of continuous maintenance executions, where n = 6 if number of processes is more than or equal to 6) in one second. The server processes are generated until the number of waiting processes specified in the MinSpareServers directive can be processed, or the number of all processes equals the number specified in the MaxClients directive. When 8 or more server processes are generated in a single maintenance, an error is logged (info level).

If the request processing ends, the state of the server process changes to the waiting state. If processes in the waiting state increase, only the number of processes specified in the MaxSpareServers directive remains during maintenance and other server processes are terminated.

(a) Notes

- Specify a large value in the StartServers directive, in the cases such as you must process a large number of requests immediately after starting or restarting the Web server.
- The number of the processes generated after the Web server starts are controlled depending upon the MaxSpareServers and the MinSpareServers directives, and hence the specified value in the StartServers directive becomes irrelevant (except

when the HWSKeepStartServers directive is set to `On`). Specify the MinSpareServers and the MaxSpareServers directives for preparing the processes in waiting state such that they can handle the sudden increase in number of requests. If error logging (info level) occurs frequently in the maintenance process, adjust the values of these directives to increase the number of waiting processes.

- If many server processes are always kept in the waiting state, concurrent connection requests from many clients can be received. However, you need to take precaution because that much server resources will be consumed.
- When the CPU is overloaded with CGI programs, you need to set a small value in the MaxClients directive to stop receiving requests. If all the processes specified in the MaxClients directive are being processed, the requests are held in queue depending upon the ListenBacklog directive specification.
- A server process ends after processing the number of requests specified in the MaxRequestsPerChild directive. However, if the MaxRequestsPerChild directive is set to `0`, the server process does not end with the request process count. If there are chances of memory leakage in the application programs created by the end-user, the specification of the MaxRequestsPerChild directive is applied.
- When an abnormal termination signal is sent to the server process (even when a failure occurs in API connection module), the process outputs information about the abnormal termination to the error log (notice level). An error log of notice level is output regardless of the LogLevel directive specification.
- When you want to always keep the number of server processes specified in the StartServers directive running regardless of the number specified in the MaxSpareServers and MinSpareServers directives, set the HWSKeepStartServers directive to `On`. If the server process count is less than the number specified in the StartServers directive, the Web server generates new processes to arrive at the specified number.
- If you stop Cosminexus HTTP Server, the files specified in the PidFile directive are deleted. However, when Cosminexus HTTP Server is forcibly stopped by something other than the program, for example, the machine is shut down without stopping Cosminexus HTTP Server, the files specified in the PidFile directive remain undeleted. Starting Cosminexus HTTP Server again might result in a failure. If the files specified in the PidFile directive are present while Cosminexus HTTP Server is not running, delete the files before restarting Cosminexus HTTP Server.

(b) Example of transition of number of processes

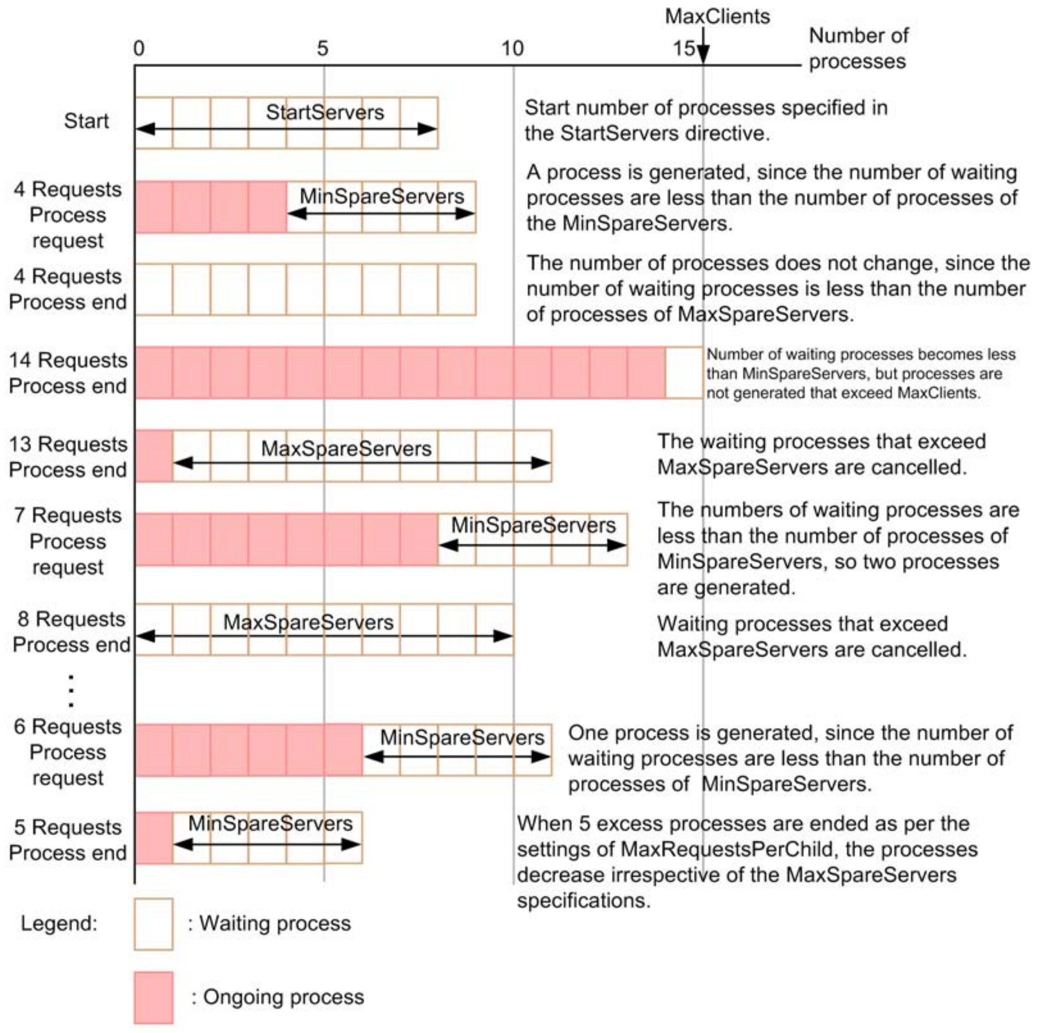
The following figure shows an example for the transition of number of processes when the specification of HWSKeepStartServers is `Off`.

- Specified values of directives (when HWSKeepStartServers is set to `Off`)

```
StartServers 8
MaxSpareServers 10
MinSpareServers 5
MaxClients 15
HWSKeepStartServers Off
MaxRequestsPerChild 10000
```

```
KeepAlive Off
```

Figure 4-2: An example of transition of number of processes (when HWSKeepStartServers is set to `Off`)


MaxClients
0
5
10
15
Number of processes
Start
StartServers
Start number of processes specified in the StartServers directive.
4 Requests Process request
MinSpareServers
A process is generated, since the number of waiting processes are less than the number of processes of the MinSpareServers.
4 Requests Process end
The number of processes does not change, since the number of waiting processes is less than the number of processes of MaxSpareServers.
14 Requests Process end
Number of waiting processes becomes less than MinSpareServers, but processes are not generated that exceed MaxClients.
13 Requests Process end
MaxSpareServers
The waiting processes that exceed MaxSpareServers are cancelled.
7 Requests Process request
MinSpareServers
The numbers of waiting processes are less than the number of processes of MinSpareServers, so two processes are generated.
8 Requests Process end
MaxSpareServers
Waiting processes that exceed MaxSpareServers are cancelled.
6 Requests Process request
MinSpareServers
One process is generated, since the number of waiting processes are less than the number of processes of MinSpareServers.
5 Requests Process end
MinSpareServers
When 5 excess processes are ended as per the settings of MaxRequestsPerChild, the processes decrease irrespective of the MaxSpareServers specifications.
Legend:
: Waiting process
: Ongoing process

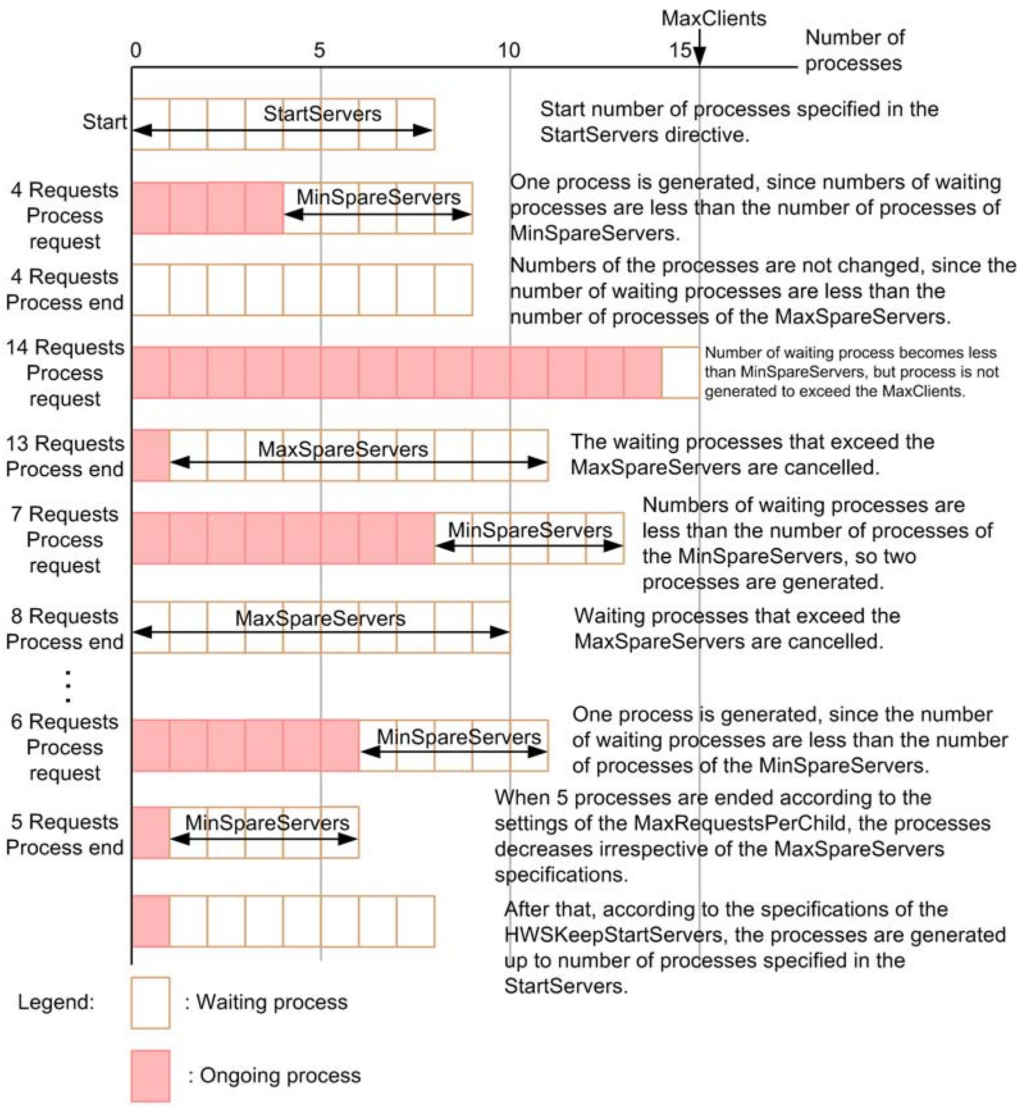
The following figure shows an example of transition of number of processes when HWSKeepStartServers is set to `On`.

- Specified values of directives (when HWSKeepStartServers is set to `On`)

```
StartServers 8
MaxSpareServers 10
MinSpareServers 5
MaxClients 15
```

```
HWSKeepStartServers On
MaxRequestsPerChild 10000
KeepAlive Off
```

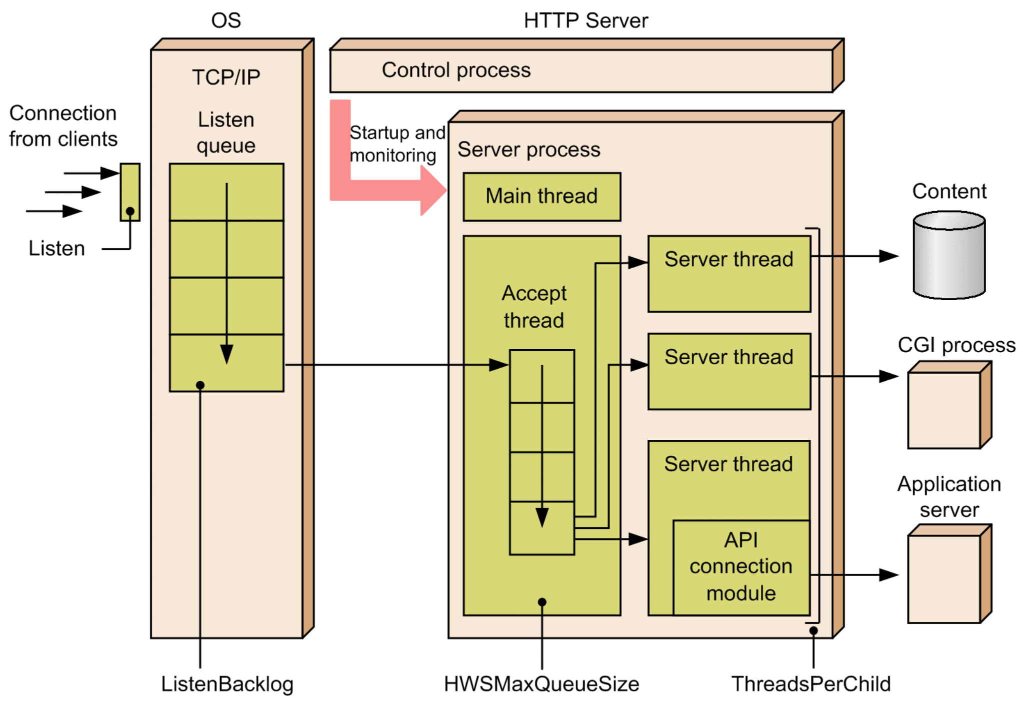
Figure 4-3: An example of transition of number of processes (when HWSKeepStartServers is set to `On`)


MaxClients
0
5
10
15
Number of processes
Start
StartServers
Start number of processes specified in the StartServers directive.
4 Requests Process request
MinSpareServers
One process is generated, since numbers of waiting processes are less than the number of processes of MinSpareServers.
4 Requests Process end
Numbers of the processes are not changed, since the number of waiting processes are less than the number of processes of the MaxSpareServers.
14 Requests Process request
Number of waiting process becomes less than MinSpareServers, but process is not generated to exceed the MaxClients.
13 Requests Process end
MaxSpareServers
The waiting processes that exceed the MaxSpareServers are cancelled.
7 Requests Process request
MinSpareServers
Numbers of waiting processes are less than the number of processes of the MinSpareServers, so two processes are generated.
8 Requests Process end
MaxSpareServers
Waiting processes that exceed the MaxSpareServers are cancelled.
6 Requests Process request
MinSpareServers
One process is generated, since the number of waiting processes are less than the number of processes of the MinSpareServers.
5 Requests Process end
MinSpareServers
When 5 processes are ended according to the settings of the MaxRequestsPerChild, the processes decreases irrespective of the MaxSpareServers specifications.
After that, according to the specifications of the HWSKeepStartServers, the processes are generated up to number of processes specified in the StartServers.
Legend:
: Waiting process
: Ongoing process

4.1.2 Architecture of Cosminexus HTTP Server process (Windows Version)

The following figure shows the architecture of Cosminexus HTTP Server process.

Figure 4-4: Cosminexus HTTP Server process architecture (Windows Version)



When you start Cosminexus HTTP Server, the control process starts. The control process starts the server process and monitors the server process operations. When the server process starts, the main thread also starts simultaneously. The main thread starts an `accept` thread to receive requests and starts the request processing server threads equal to the number specified in the ThreadsPerChild directive.

The OS accepts TCP connection of a client, through the IP address and port specified in the Listen directive, and reserves the connection in the Listen queue of OS. You can specify the size of Listen queue in the ListenBacklog directive. The TCP connection that cannot be stored in the Listen queue is not established. The `accept` thread accepts the TCP connection from the Listen queue and registers the connection in the request queue with the size specified in the HWSMaxQueueSize directive. One of the server threads picks up this connection, and then receives and processes the HTTP request. If the TCP connection cannot be stored in the request queue as the TCP connection exceeds the HWSMaxQueueSize directive value, the `accept` thread closes the TCP connection.

You cannot increase or decrease the number of server threads.

4.1.3 Operation management

This section describes the principles of persistent connection operation and time monitoring functionality required to manage the operational conditions of server process (server thread for Windows Version).

(1) Persistent connection (KeepAlive)

In persistent connection (KeepAlive) functionality, the TCP connection does not disconnect even after responding to client request, and waits for the next request from the same client.

You can use this functionality by setting the KeepAlive directive to `On`, and when the client is responding. When the client sends multiple requests continuously, the response time can be reduced, since the TCP connection is not disconnected.

When waiting for the next request, the server processes are occupied in the client, but you can set the waiting time with the KeepAliveTimeout directive. In the MaxKeepAliveRequests directive, specify how many times a singe client can process the requests with a persistent connection.

(2) Time monitoring

You can monitor the time based on the value set in the Timeout directive in the following cases:

- When receiving requests from the client (after the connection is established and the HTTP protocol is received)
- When sending a response to the client
- When sending a request to the CGI program
- After sending a request to the CGI program until a response is received
- When receiving a response from the CGI program
- The wait time after receiving a response from a CGI program until the I/O pipe is closed
- When sending a request to a back-end server if a reverse proxy is used
- From the time a request is sent to a back-end server until a response is received if a reverse proxy is used
- When receiving a response from a back-end server if a reverse proxy is used

(3) Monitoring the control process

If you monitor the process of the ID that is output to the file specified in the PidFile directive, you can monitor Cosminexus HTTP Server control process. The process name (execution program name) to be monitored is `httpsd.exe` in the Windows version, and `httpsd` in the UNIX version.

When you monitor the control process, always confirm that the process of the ID stored in the file specified with PidFile directive is Cosminexus HTTP Server process. To confirm that the process is a Cosminexus HTTP Server process, make sure that the execution program name of the process is `httpsd.exe` in Windows version, and `httpsd` in UNIX version.

4.2 Collecting logs

For the meaning of output messages, see *23. Messages output by Web Server (Cosminexus HTTP Server)* and *26. KAWS (Audit Log Messages Output by Cosminexus HTTP Server)* in the manual *uCosminexus Application Server Messages*.

4.2.1 Log types

The following table describes the types of log.

Table 4-1: Types of log

Types of log	Directives to be specified	Functionality
Access log	TransferLog	• Collect the log in default format. • You can use the rotatelogs or rotatelogs2 program to divide the log periodically and quantitatively. • You can change the log file format with the LogFormat directive.
	CustomLog	• Collect the log in customized format. • You can use the rotatelogs or rotatelogs2 program to divide the log periodically and quantitatively. • You can specify the format defined with the LogFormat directive in the CustomLog directive.
Error log	ErrorLog	• Collect the log of error message when an error occurs. • You can use the rotatelogs or rotatelogs2 program to divide the log periodically and quantitatively. • You can specify the log level to be collected with the LogLevel directive. • When the HWSRequestLog directive is not specified, module trace information can be collected. For details on the module trace, see *4.2.6 Collecting the module trace*.
	ScriptLog	• Collect the CGI script error log.

Types of log	Directives to be specified	Functionality
Request log	HWSRequestLog	• Collects the request log. The following trace information can be collected as the request log: Module trace information Module trace information is collected when module functions and CGI programs are executed. For details on module trace information, see *4.2.6 Collecting the module trace*. Request trace information Request trace information is collected when the request process starts and is completed. For details on request trace information, see *4.2.7 Collecting request trace information*. I/O filter trace information I/O filter trace information is collected when the input and output filter function implemented in the module is executed. For details on I/O filter trace information, see *4.2.8 Collecting I/O filter trace information*. • You can use the rotatelogs or rotatelogs2 program to divide the log periodically and quantitatively.
Process ID log	PidFile	• Collect the control process ID log.
Event log	None	• Record the error that occurs when you start the server from a service (Windows Version).
Internal trace	HWSTraceLogFile	• Output the trace information of shared memory.
Shared memory ID log	HWSTraceIdFile	• Store the shared memory ID.
Core file#	CoreDumpDirectory	• Specify a directory where the core dump will be output when a failure occurs in Cosminexus HTTP Server (UNIX Version).
	SSLCacheServerRunDir	• Specify a directory where the core dump is output when a failure occurs in the gcache server (used in SSL) (UNIX Version).

#

The log is output when you specify the settings to output the core file in the OS.

For details on the setting method, see the respective OS manual.

If the size of the access log, error log or request log exceeds 2 GB, Cosminexus HTTP Server might abnormally end or not restart. Back up the log files periodically, or specify settings so that the sizes of the log files do not exceed 2 GB, as noted in *4.2.3*

Dividing logs (rotatelogs program) and *4.2.4 Reusing the log files by wrapping around (rotatelogs2 program)*.

4.2.2 How to collect logs

This subsection describes how to collect the access log, error log, process ID log, and request log.

(1) Access log

(a) Access log in the default format

Specify the TransferLog directive to acquire the log.

The following is an example of access log in the default format:

```
Client host name Δ Identification information of the client Δ Client user name Δ Access
time Δ "request line" Δ Status code Δ Number of sent bytes
```

Legend:

Δ : Space

(Output example)

```
172.17.40.30 - - [25/Dec/2000:16:23:59 +0900] "GET / HTTP/1.0" 200 3546
```

(b) Access log of custom format

Specify the CustomLog directive and collect the log. There are two methods to specify format:

- Specify the format directly in the CustomLog directive

  (Example)

  ```
  CustomLog logs/access.log "%h %l %u %t \"%r\" %>s %b"
  ```

- Define a label name for the format with the LogFormat directive, and specify this label name in the CustomLog directive

  (Example)

  ```
  LogFormat "%h %l %u %t \"%r\" %>s %b" common
  CustomLog logs/access.log common
  ```

(2) Error log

(a) Error message log

Specify the ErrorLog directive and collect the log. Specify the level of the errors to be collected with the LogLevel directive.

(b) The CGI script error log

Specify the ScriptLog directive and collect the CGI script error log.

(3) Process ID log

Specify the PidFile directive and collect the control process ID log.

(4) request log

Specify the HWSRequestLog and the HWSRequestLogType directives to collect the request log. The request log is a generic name that refers to module trace information, request trace information, and I/O filter trace information.

For details on module trace information, see *4.2.6 Collecting the module trace*. For details on request trace information, see *4.2.7 Collecting request trace information*. For details on I/O filter trace information, see *4.2.8 Collecting I/O filter trace information*.

(5) Locations to which trace information is output

(a) Module trace information output destination

Module trace information is output to either the error log or the request log. The specification of directives determines which log is used for outputting module trace information. The following table lists the module trace information output destinations and conditions.

Table 4-2: Module trace information output destinations and conditions

Output destination	Output conditions
Request log	The HWSRequestLog directive is specified, and either module-info or module-debug is specified in the HWSRequestLogType directive.
Error log	The HWSRequestLog directive is not specified, and either info or debug is specified in the LogLevel directive.

For details on module trace information, see *4.2.6 Collecting the module trace*.

(b) Output destination of request trace information and I/O filter trace information

Request trace information and I/O filter trace information are output to the request log.

When the HWSRequestLog directive is specified and the HWSRequestLogType directive satisfies the output conditions, the trace information is output to the request log. For details on the output conditions of the HWSRequestLogType directive, see *4.2.7 Collecting request trace information* and *4.2.8 Collecting I/O filter trace information*.

4.2.3 Dividing logs (rotatelogs program)

You can divide access logs and error logs by a specific time period (for example, every 24 hours) and output the logs to multiple files. The rotatelogs program can be specified in the following directives:

- CustomLog directive
- ErrorLog directive
- HWSRequestLog directive
- TransferLog directive

Specify the program in the following format.

(1) Format

```
rotatelogs prefix-for-split-log-file log-splitting-time-interval [-fnum number-of-files] [-diff
time-difference-from-GMT]
```

(2) Parameters

- *Prefix-for-split-log-file*

  This parameter specifies the prefix of split log file with an absolute path.

  Collect the log in the file called '*Prefix.nnnnnnnnnn*' file.

  nnnnnnnnnn: Displays the log collection start time. The log collection time refers to a value displayed in the following format:

  > (The value rounded off after the decimal point of (number of seconds taken to output the log, when the starting time is January 1, 1970 at 00:00:00 (GMT: Greenwich Mean Time) ÷ log splitting time interval)) × log splitting time interval

- *log-splitting-time-interval* ~ ((1-31536000))

  This parameter specifies the time interval to collect one log file in seconds. The log is collected in a new file every time when the specified time elapses.

- -fnum *number-of-files* ~ ((1-256))

  This parameter specifies the maximum number of split log files. If the number of split files exceeds the maximum number specified with this parameter, the files starting from the oldest file are deleted. If you do not specify this parameter, the log files are not deleted.

- `-diff` *time-difference-from-GMT* `((-1439-1439))`

  This parameter specifies the time offset (in minutes) from GMT for splitting the log files. If you do not specify the standard time, or if you specify `0`, January 1, 1970 at 00:00:00 (GMT) is the standard time. When difference in local time with respect to GMT is n hours, and if m hours 0 minute 0 second of the local time is the standard time, specify (n - m) × 60. If 0 hours 0 minute 0 second of JST is the standard time, specify 540 in (+ 9 - 0) × 60.

(3) How to use

Use the `rotatelogs` after specifying " | Program name" format in the TransferLog, CustomLog, and the ErrorLog directives. The `rotatelogs` splits the log file periodically into separate files to collect the logs.

(Example) Windows version

This example explains how to split the access log after every 24 hours, and to collect the log in the file <*Cosminexus-installation-directory*>\httpsd\logs\access.nnnnnnnnnn on Windows. The following is the specification, where you set the splitting time as per the Japan Time, and split the log file at every 0 hrs of Japan time:

```
TransferLog "|\"\"<Cosminexus-installation-directory>/httpsd/sbin/
rotatelogs.exe\" \"<Cosminexus-installation-directory>/httpsd/logs/access\"
86400 -diff 540\""
```

Log file name:
<*Cosminexus-installation-directory*>\httpsd\logs\access.nnnnnnnnnn

Log splitting time interval: 86400 seconds (= 24 hours)

(Example) UNIX version

This example describes how to split the access log after every 24 hours, and to collect the log in /opt/hitachi/httpsd/logs/access.nnnnnnnnnn file on UNIX. The following is the specification, where you set the splitting time as per the Japan Time, and split the log file at every 0 hrs of Japan time:

```
TransferLog "|/opt/hitachi/httpsd/sbin/rotatelogs /opt/hitachi/httpsd/logs/
access 86400 -diff 540"
```

Log file name: /opt/hitachi/httpsd/logs/access.nnnnnnnnnn

Time interval to divide the log: 86400 seconds (= 24 hours)

(4) Notes

(a) Notes for the UNIX version

- The rotatelogs2 program does not end the process even if a SIGTERM, SIGUSR1 or SIGHUP signal is received, but ends the process when the control process and server process end.

(b) Notes for the Windows version

- When you start the Web server as a service, the control process log is not collected.
- You cannot delete the log file until the process that has opened the file still exists. As a result, the number of files remaining might be more than the value specified for `-fnum`. For example, control process log files are deleted only when the control process ends.
- If an incorrect argument is present at startup, the Web server starts but the rotatelogs program does not start. The rotatelogs program outputs messages that contain the following attributes to the event log:
    - Type: Error
    - Source: CosminexusHTTPServer
    - Class: none
    - Event: 3299
    - Description: rotatelogs.exe: (message)

      For the meaning of output messages, see *23.9(6) rotatelogs program* in the manual *Cosminexus Application Server Messages*.

  When this message is displayed, follow the instructions in the message and specify valid arguments before restarting the server. Note that the same message might be output multiple times.

(c) Common notes for the UNIX and Windows versions

- For controlling the log file as per the `-fnum` parameter specification when you restart the Web server, if there is a change in the directory name or the log file prefix, the log file that is previously extracted is not deleted. In such cases, delete the log file based on the operation.
- When the specified time interval to split the log elapses since the Web server is started or restarted, and if the number of files matching to the prefix for split log files exceeds the value specified in the `-fnum` parameter, the files are deleted from the oldest files onwards.
- Specify the *prefix-for-split-log-files* with an absolute path.
- If an argument is not specified correctly in the rotatelogs program for TransferLog, CustomLog, ErrorLog, and HWSRequestLog directives, the rotatelogs program fails to start but the Web server starts properly. In this case, logs are not output. When you specify the rotatelogs program for TransferLog,

CustomLog, ErrorLog, and HWSRequestLog directives, make sure that log files are created and the files are divided as you intended.

4.2.4 Reusing the log files by wrapping around (rotatelogs2 program)

You can split the access log and error log based on the log file size and output them to multiple log files by wrap around. The rotatelogs2 program can be specified in the following directives:

- CustomLog directive
- ErrorLog directive
- HWSRequestLog directive
- TransferLog directive

Specify the program in the following format.

(1) Format

```
rotatelogs2 Log-file-prefix-name Log-file-size Number-of-log-files
```

(2) Parameters

- *Log-file-prefix-name*

  This parameter specifies the prefix of output log file with an absolute path.

  The output log file name is '`Prefix.nnn`'.

  `.nnn` is from `.001` up to the value specified in the number of log files.

  If 'Number of log files' is considered as nnn files, the last modified file in nnn files when Cosminexus HTTP Server is started, becomes the current log file. The log files are categorized by adding an extension `.001~.nnn` to the file name. When the extension of the current log file is `.mmm`, and if the current log file is full, then `.mmm+1` log file is cleared and output. When `.mmm` matches with `.nnn`, the next log file is output with `.001` extension.

  For the Windows version, the *prefix*`.index` file for index number storage is created. The *prefix*`.index` file, which is used for `.`*nnn* management, is created when the rotatelogs2 utility starts, and is deleted when the rotatelogs2 program stops. However, the *prefix*`.index` file might not be deleted because of a startup error. This does not affect subsequent Web server operation.

- *Log-file-size* ~ ((1-2097151))

  This parameter specifies the maximum size for a log file (unit: KB).

When the log file is output, and if the size exceeds the maximum size of log file, the rotatelogs2 utility clears the next log file and the output is continued.

- *Number-of-log-files* ~ ((1-256))

  This parameter specifies the maximum number of log files to be output.

  When the log file exceeds the maximum size and moves to the next file, and if the extension of the processed log files is same as the maximum number of files, reuse the from the file with extension `.001`.

(3) How to use

Specify the rotatelogs2 utility by using the `|`*program-name* format in directives.

Example: Collecting a maximum number of five error log files for every 4,096 KB

```
ErrorLog "|\"\"<Cosminexus-installation-directory>/httpsd/sbin/
rotatelogs2.exe\" \"<Cosminexus-installation-directory>/httpsd/logs/
errorlog\" 4096 5\""
```

The log is output in the sequence of errorlog.001 ~ errorlog.005. If the errorlog.005 exceeds 4,096 KB, errorlog.001 is cleared and output is continued. If these log files already exist when Cosminexus HTTP Server starts, the last updated log file is output. If the size of this log file already exceeds 4,096 KB, the next log file is cleared and the log output is continued. When the file size does not exceed 4,096 KB, the output is continued to the same file.

(4) Notes

(a) Notes for the UNIX version

- The rotatelogs2 program does not end the process even if a SIGTERM, SIGUSR1 or SIGHUP signal is received, but ends the process when the control process and server process end.

(b) Notes for the Windows version

- Start the Web server as a service. If you do not start the Web server as a service, the wrong log file might be cleared when you stop or restart the Web server.
- Do not edit or delete the index number storage file while the rotatelogs2 program is running. If you edit the index number storage file, logs might not be output correctly
- If a file with the same name as the prefix.index file for index number storage already exists when the Web server starts, the file is overwritten.
- If an incorrect argument is present at startup, the Web server starts but the rotatelogs2 program does not start. The rotatelogs2 program outputs messages

that contain the following attributes to the event log.

- Type: Error
- Source: CosminexusHTTPServer
- Class: none
- Event: 3299
- Description: rotatelogs2.exe: (message)

  For the meaning of output messages, see *23.9(7) rotatelogs2program* in the manual *Cosminexus Application Server Messages*.

When this message is displayed, follow the instructions in the message and specify valid arguments before restarting the server. Note that the same message might be output multiple times.

(c) Common notes for the UNIX and Windows versions

- Specify the prefix name of log file with an absolute path.
- When Cosminexus HTTP Server starts, the last modified log file is targeted for output and if a wrong file is updated, a wrong file is output.
- When specifying log file size, do not specify a size that is so small that multiple log files exceed the specified size within same seconds. If you specify such a small size, proper rotation does not take place and the latest log file is not output.
- In the configuration file, do not specify the same prefix for log files at multiple places. When you specify at multiple places, all files except the latest file are output and correct rotation does not take place.
- If an argument is not specified correctly in the rotatelogs2 program for TransferLog, CustomLog, ErrorLog, and HWSRequestLog directives, the rotatelogs2 program fails to start but the Web server starts properly. In this case, logs are not output. When you specify the rotatelogs2 program for TransferLog, CustomLog, ErrorLog, and HWSRequestLog directives, make sure that the log files are created and the files are divided as you intended.

4.2.5 Converting the IP address of log file into a host name (logresolve command)

The logresolve command converts the IP address of the access log file (that has IP address at the beginning of the record) into the host, and outputs to the new log file. The conversion rule depends on the reverse lookup of host name.

(1) Format

```
logresolve [-s file-name] [-c] Access-log-file-name New-log-file-name
```

(2) Parameters

- -s *file-name*

  This parameter specifies the output file name that contains the converted IP address. The following information is output to this file:

  - IP address and the host name after conversion
  - IP addresses that cannot be converted
  - Count of converted IP addresses

- -c

  Use this option to check whether the host name after conversion matches the IP address before conversion.

- *Access-log-file-name*

  This parameter specifies the input log file name. Reverse the look up of the host name from the IP address mentioned in the input file. The IP address must be at the top of the record. If an attempt to retrieve the host name fails, the IP address is output to the new log file.

- *New-log-file-name*

  This parameter specifies a file name that outputs the access log with IP address converted to the host.

(3) How to use

Convert the IP address of access log stored in the `logs\access.log`, into the host name.

Access log file: logs\access.log

New log file: logs\new_access.log

```
logresolve < logs\access.log > logs\new_access.log
```

4.2.6 Collecting the module trace

The Web server consists of multiple modules#, and these modules consist of multiple functions to be executed at a specific time. Module trace information indicates the trace information collected when functions in modules and CGI programs are executed. The specifications for module trace information collection, such as the destination of the module trace information collection, is changed depending on whether the HWSRequestLog directive is specified.

#: Modules refer to both external modules, which are dynamically embedded and used by using the LoadModule directive in Web server, and internal modules, which are included in the httpsd execution file.

(1) Trace target

The following table describes the trace target of module trace:

Table 4-3: Trace target of module trace

Trace target	Triggers
Modules	Modules consist of multiple functions. These functions are classified into functions for initialization processes and functions for request handling processes. The trace information of functions for request handling processes is collected.
CGI programs	Collect the trace when the CGI programs are running.

(2) How to collect module trace information

When the HWSRequestLog directive is not specified, module trace information is collected and placed in the file specified in the ErrorLog directive according to the specification of the LogLevel directive.

When the HWSRequestLog directive is specified, module trace information is collected and placed in the file specified in the HWSRequestLog directive according to the specification of the HWSRequestLogType directive.

Note

> When logs are collected and placed in the file specified in the ErrorLog directive, the file can be split by virtual hosts. If logs are collected and placed in the file specified in the HWSRequestLog directive, the file cannot be split by virtual hosts.

(3) Collection level

The level of the module trace information to be collected can be changed according to the LogLevel directive or the HWSRequestLogType directive. The contents of the trace information collected at each level are as described below.

(a) info level

The trace information for external modules and CGI programs which might cause a failure is collected.

When you specify either info in the LogLevel directive or module-info in the HWSRequestLogType directive, the trace information is collected.

(b) debug level

In addition to info-level trace information, trace information for internal modules, which run for each request, are collected.

When you specify either debug in the LogLevel directive or module-debug in the HWSRequestLogType directive, trace information is collected.

(4) Trace format

The following are output items of module trace:

Note that hereafter, the *Servere-process-ID* will be the *Server-thread-ID* for Windows.

(a) Modules

- *When log is output to the info level*

Call time

`[Time]` Δ `[info]` Δ `hws` Δ `:` Δ `module` Δ `-->` Δ `(`*module-file-name*`[`*function-offset*`])(`*server-process-ID*`)`

Return time

`[Time]` Δ `[info]` Δ `hws` Δ `:` Δ `module` Δ `<--` Δ `(`*module-file-name*`[`*function-offset*`])(`*server-process-ID*`)(`*result-code*`)`

Legend:

Δ : Space

The following table describes the relationship between the *function offset* and the functions:

Table 4-4: Relationship between function offsets and the functions

Function offset	Function name	Meaning
[0]	create_request	Executing the request start process.
[1]	post_read_request	Executing the process after reading the request.
[2]	quick_handler	Executing the process before changing the requested URL.

Function offset	Function name	Meaning
[3]	translate_name	Executing the process to change the actual file name from the requested URL.
[4]	map_to_storage	Executing the request process that does not involve disk access.
[5]	header_parser	Executing the process to analyze the request header.
[6]	access_checker	Checking the access permission by the host name, and the IP address for the URL requested from an authenticated user.
[7]	check_user_id	Checking the user ID.
[8]	auth_checker	Checking the access permission (Require) for the URL requested from an authenticated user.
[9]	type_checker	Checking the MIME type.
[10]	fixups	Executing the process before request execution.
[11]	insert_filter	Executing the filter insertion process.
[12]	handler	Executing handler.
[13]	insert_error_filter	Executing the process before responding to error.
[14]	log_transaction	Executing the log output process.
[15]	error_log	Executing the process after error log output.
[16]	get_suexec_identity	Executing the process to acquire the user information.

(Output example)

```
[Fri Jul 15 17:29:43 2005] [info] hws : module --> (mod_example.c[1])(1864)
[Fri Jul 15 17:29:43 2005] [info] hws : module <--
(mod_example.c[1])(1864)(-1)
```

- *When log is output to debug level*

Call time

`[Time]`Δ`[debug]`Δ*file-name*`(`*line-number*`):`Δ`hws`Δ`:`Δ`module`Δ`-->`Δ`(`*module-file-name*`[`*function-offset*`])(`*server-process-ID*`)`

Return time

```
[Time] Δ [debug] Δ file-name(line-number): Δ hws Δ : Δ module Δ <-- Δ (module-file-na
me[function-offset])(server-process-ID)(result-code)
```

Legend:

Δ : Space

(Output example)

```
[Fri Jul 15 17:29:43 2005] [debug] request.c(69): hws : module -->
(mod_alias.c[3])(1864)
[Fri Jul 15 17:29:43 2005] [debug] request.c(69): hws : module <--
(mod_alias.c[3])(1864)(-1)
```

(b) CGI program

- When module trace information is output at the info level

Call time

```
[Time] Δ [info] Δ hws Δ : Δ cgi Δ --> Δ (exec=cgi-file-name)(argv0=execution-program
-name)(args=arguments#)(server-process-ID)(cgi-process-ID)
```

#: The argument from args is displayed only when the specified query is followed by a + (plus) and not by = (is equals to), as in the case of GET. /cgi-bin/ isindex?aaa+bbb+ccc HTTP/1.0.

Return time

```
[Time] Δ [info] Δ hws Δ : Δ cgi Δ <-- Δ (exec=cgi-file-name)(argv0=
execution-program-name)(server-process-ID)(cgi-process-ID)
```

Legend:

Δ : Space

(Output example)

```
[Fri Jul 15 19:48:08 2012] [info] hws : cgi -->
(exec=<Cosminexus-installation-directory>/httpsd/cgi-bin/
isindex)(argv0=isindex)(args=aaa+bbb+ccc)(1784)(1144)
[Fri Jul 15 19:48:08 2012] [info] hws : cgi <--
(exec=<Cosminexus-installation-directory>/httpsd/cgi-bin/
isindex)(argv0=isindex)(1784)(1144)
```

(5) How to use

(a) Usage example

An example to output info-level module trace information and request trace information to the request log is as follows:

```
HWSRequestLogType module-info request
HWSRequestLog logs/hwsrequest.log
```

(b) Example of trace for abnormal exit

- When a error occurs in an external module

```
[Fri Jul 15 10:29:29 2005] [info] hws : module --> (mod_example.c[1])(1800)
[Fri Jul 15 10:29:30 2005] [notice] Parent: child process exited with status
3221225477 -- Restarting.
```

An abnormality has occurred in the function that corresponds to mod_example.c[1] that is the post_read_request.

Check the function that corresponds to the mod_example.c[1].

- When there is no response from the CGI program

```
[Fri Jul 15 19:48:39 2012] [info] hws : cgi -->
(exec=<Cosminexus-installation-directory>/httpsd/cgi-bin/
test-sleep)(argv0=test-sleep)(1800)(2276)
[Fri Jul 15 19:48:49 2012] [info] [client 192.168.1.1] Premature end of script
headers: test-sleep
[Fri Jul 15 19:48:49 2012] [info] hws : cgi <--
(exec=<Cosminexus-installation-directory>/httpsd/cgi-bin/
test-sleep)(argv0=test-sleep)(1800)(2276)
```

In the CGI program, check this function as timeout occurs in the test-sleep process.

4.2.7 Collecting request trace information

Request trace information indicates the trace information collected for the following cases:

- When a request process starts
- When a request process is completed
- When a request line is received with a KeepAlive connection
- When the connection is removed after a request process starts but before the request line is received.

The request trace information collection functionality can be used when the HWSRequestLog directive is specified and request is specified in the HWSRequestLogType directive. Request trace information is useful for checking whether the Web server receives a request when a failure occurs.

(1) Trace information format

The items below are output for request trace information.

Note that hereafter the *server-process-ID* indicates the *server-thread-ID* for the Windows version.

■ When a request process starts

```
[Time]Δclient Δ:Δhws Δ-->Δ(client-IP-address:port-number,server-IP-address:port-number[A]
)(server-process-ID)
```

■ When a request process is completed

```
[Time]Δclient Δ:Δhws Δ<--Δ(client-IP-address:port-number,
server-IP-address:port-number[R])(server-process-ID)
```

■ When the next request line is received with the KeepAlive connection

```
[Time]Δclient Δ:Δhws Δ-->Δ(client-IP-address:port-number,server-IP-address:
port-number[K])(server-process-ID)
```

- When the connection is removed after a request process starts but before the request line is received

```
[Time] Δ client Δ : Δ hws Δ <-- Δ (client-IP-address: port-number,server-IP-address:
port-number[X])(server-process-ID)
```

Legend:

Δ : Space

(Output example)

```
[Tue Nov 21 15:18:40 2006] client : hws --> (192.168.2.1:5245,192.168.1.1:80[A])(1716)
[Tue Nov 21 15:18:41 2006] client : hws <-- (192.168.2.1:5245,192.168.1.1:80[R])(1716)
```

4.2.8 Collecting I/O filter trace information

I/O filter trace information indicates the trace information collected when the input and output filter function implemented in a module is executed.

The I/O filter trace information collection functionality can be used when the HWSRequestLog directive is specified and filter is specified in the HWSRequestLogType directive. I/O filter trace information is useful for isolating an error that occurred in a module filter. Due to the large amount of data output for I/O filter trace information, we do not recommend using it for anything aside from debugging.

(1) Trace information format

The items below are output for I/O filter trace information.

Note that hereafter the *server-process-ID* indicates the *server-thread-ID* for the Windows version.

- For an input filter call

```
[Time] Δ hws Δ : Δ in-filter# Δ --> Δ (filter-name[filter-type-number])
(server-process-ID)
```

- For an input filter return

```
[Time] Δ hws Δ : Δ in-filter# Δ <-- Δ (filter-name[filter-type-number])
(server-process-ID)(returned-value)
```

Legend:

Δ : Space

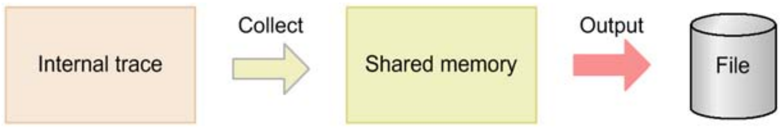
#: For the output filter, in-filter is changed to out-filter.

(Output example)

```
[Tue Nov 21 15:18:40 2006] hws : in-filter --> (core_in[60])(1716)
[Tue Nov 21 15:18:40 2006] hws : in-filter <-- (core_in[60])(1716)(0)
```

4.2.9 Collecting the internal trace (hwstraceinfo command)

When an application program is executed and a request is received, the events that occur in the system are collected as internal trace. The internal trace is output once to the shared memory, and then it is output to the file as per the specification in the directive or command.



(1) Collecting trace information

Internal traces are collected in the shared memory when various events occur in the Web server. The memory identifiers of shared memory are stored in the file specified in the HWSTraceIdFile directive.

(2) How to output to a file

The internal trace that is collected in the shared memory is output to a file when the server process terminates abnormally or when the hwstraceinfo command is executed. When the server process terminates abnormally, the trace is output to the file specified in the HWSTraceLogFile directive.

Specify the memory identifier and file name of output destination, in the hwstraceinfo command. For the UNIX version, only the user specified in the User directive or the superuser can execute the hwstraceinfo command. For the Windows version, only the user with administrative permission can execute the hwstraceinfo utility.

The internal trace information output file size is as follows:

For the UNIX version

Output size of the ps -efl command + output size of the vmstat command + output size of the ipcs -a command + 7 KB × MaxClient value

For the Windows version

7 KB × ThreadPerChild value

(3) hwstraceinfo command

This section describes how to specify the hwstraceinfo command.

(a) Format

```
hwstraceinfo -i shared-memory-identifier {-l file-name|-r}
```

(b) Parameters

- -i *shared-memory-identifier*

  This parameter specifies the shared memory identifier that is output to the file specified in the HWSTraceIdFile directive.

- -l *file-name*

  This parameter specifies the file that outputs the trace corresponding to the shared memory identifier specified with `-i`.

- -r

  This parameter releases the shared memory allocated to the shared memory identifier specified in `-i`. In UNIX version, the shared memory for trace remains even if the Web server stops. Use this parameter to release the remaining shared memory. In Windows version, the shared memory for trace is released when you terminate the Web server, so this parameter is not provided.

(c) Usage example

The following is an example to output the trace corresponding to the shared memory identifier 1800_1133780652_0, to the traceinfo.log file:

```
hwstraceinfo -i 1800_1133780652_0 -l traceinfo.log
```

(4) Points to be noted when releasing shared memory and restarting (UNIX Version)

To retain trace information, the Web server does not release the shared memory even when the Web server stops. The shared memory is reused when the server restarts.

When you stop the server and restart it later, the Web server releases the shared memory once and then restores it depending upon the file value specified in the HWSTraceIdFile directive. However, in the following cases, the shared memory used earlier cannot be released:

- When the same user does not restart the server (the User directive value or the

Group directive value has changed)

- The value of the HWSTraceIdFile directive has changed
- The file specified in the HWSTraceIdFile directive is deleted

When you release the shared memory, execute the hwstraceinfo command in which `-r` is specified.

4.2.10 Functionality of maintenance information collection (hwscollect command)

When the Web server terminates abnormally and does not respond, the maintenance personnel requires documents like core dump, error log, and access log to check the cause of failure. You can collect the documents required to check the failure by the hwscollect command, in a single batch. The hwscollect command is valid only on UNIX version.

You need to execute the hwscollect command with root permission.

(1) Format

```
hwscollect directory-where-the-collection-information-is-output [-f definition-file-name]
```

(2) Parameters

- *directory-where-the-collection-information-is-output*

  This parameter specifies the directory where the collected information is output as a tar archive file.

  Name of the archive file is `HWSyyyymmddhhmmss.tar`. Here, yyyymmdd is the date when you start hwscollect, and hhmmss is the local time limited to 24 hours from the start of hwscollect.

- -f *definition-file-name*

  This parameter specifies the `hwscollect.conf` file. Specify with an absolute path or relative path from the current directory.

(3) How to use

The following is the method to use this utility when Cosminexus HTTP Server is installed with a standard configuration:

```
/opt/hitachi/httpsd/maintenance/hwscollect /tmp
```

(4) Setting the configuration file

Define the operations of hwscollect in `hwscollect.conf` file. Delimit the keywords and the value with a space, and code the `hwscollect.conf`. Note that the keywords are not case sensitive. Line which begins with a # symbol is a comment. Specify all the file names with an absolute path. The following table describes the specifications and keywords of configuration file:

Table 4-5: Specifications and keywords of configuration file

Keyword	Value to be specified	Specification	Multiple specifications	Wild card
ServerRoot	Specify the ServerRoot directive value of `httpsd.conf`.	Mandatory	N	N
conf	Specify the file name of `httpsd.conf`.	Mandatory	N	N
trcinfo	Specify the directory in which hwstraceinfo command exists.	Mandatory	N	N
trcid	Specify the file name specified in the HWSTraceIdFile directive of the `httpsd.conf`.	Mandatory	N	N
PidFile	Specify the file name specified in the PidFile directive of `httpsd.conf`.	Mandatory	N	N
CORE	Specify the CoreDumpDirectory directive value and SSLCacheServerRunDir directive value of `httpsd.conf`.	Optional	Y	Y
LOG	Specify the file name that specifies the log in the ErrorLog directive, the TransferLog directive, and the CustomLog directive, which are in `httpsd.conf`.	Optional	Y	Y
FILES	Specify any other file useful for defect analysis in addition to the above-mentioned files.	Optional	Y	Y

Legend:

Y: Can specify.

N: Cannot specify.

(5) Example to specify a configuration file

The following is an example of configuration file specification:

```
ServerRoot /opt/hitachi/httpsd
conf /opt/hitachi/httpsd/conf/httpsd.conf
trcinfo /opt/hitachi/httpsd/sbin/
trcid /opt/hitachi/httpsd/logs/hws.trcid
PidFile /opt/hitachi/httpsd/logs/httpd.pid
CORE /opt/hitachi/httpsd/logs/core*
LOG /opt/hitachi/httpsd/logs/error*
LOG /opt/hitachi/httpsd/logs/access*
LOG /opt/hitachi/httpsd/logs/hws.trclog*
LOG /opt/hitachi/httpsd/logs/hwsrequest*
```

(6) Disk usage

- File for temporary usage

  200 KB + (7 KB × MaxClients value)
- `tar` file to output collected information

  core file size + log file size + temporary file size

(7) Notes

- Reserve some empty space in the directory where the collected information is output, as the maintenance information archive file that includes the core file will be created in the directory.
- Create the output file and the temporary file in the directory where the collected information is output. Set the directory as writable where the collected information will be output.
- If you specify a directory for CORE, LOG, and other FILES, all the files are collected under this specified directory. You need to take care if you specify a top-level directory such as the root directory, as the directory may collect a large volume of unnecessary information.

4.3 Setting up a virtual server machine (virtual host)

Virtual host represents one server machine as multiple server machines. The two methods to set up a virtual host are as follows:

- Virtual server based on server name (Name-Based Virtual Hosts)
- Virtual server based on IP address (IP-Based Virtual Hosts)

(1) Virtual host based on server name

Server name-based virtual host defines multiple host names for one IP address in a server such as the DNS server and when clients access this host name, it appears as multiple hosts. You need not set multiple network interfaces. In the server name-based virtual host, you cannot build hosts for a combination of SSL and non-SSL, or for various SSLs. When you build hosts for these combinations, build by IP address-based virtual hosts.

Example: Open a port on one Web server machine (IP address: 172.17.40.10), and switch hosts according to Web browser requests.

If the request from Web browser is http://www1.xxx.soft.hitachi.co.jp/, see <*Cosminexus-installation-directory*>*/httpsd/htdocs1/index.html* (when DirectoryIndex is specified as `index.html`).

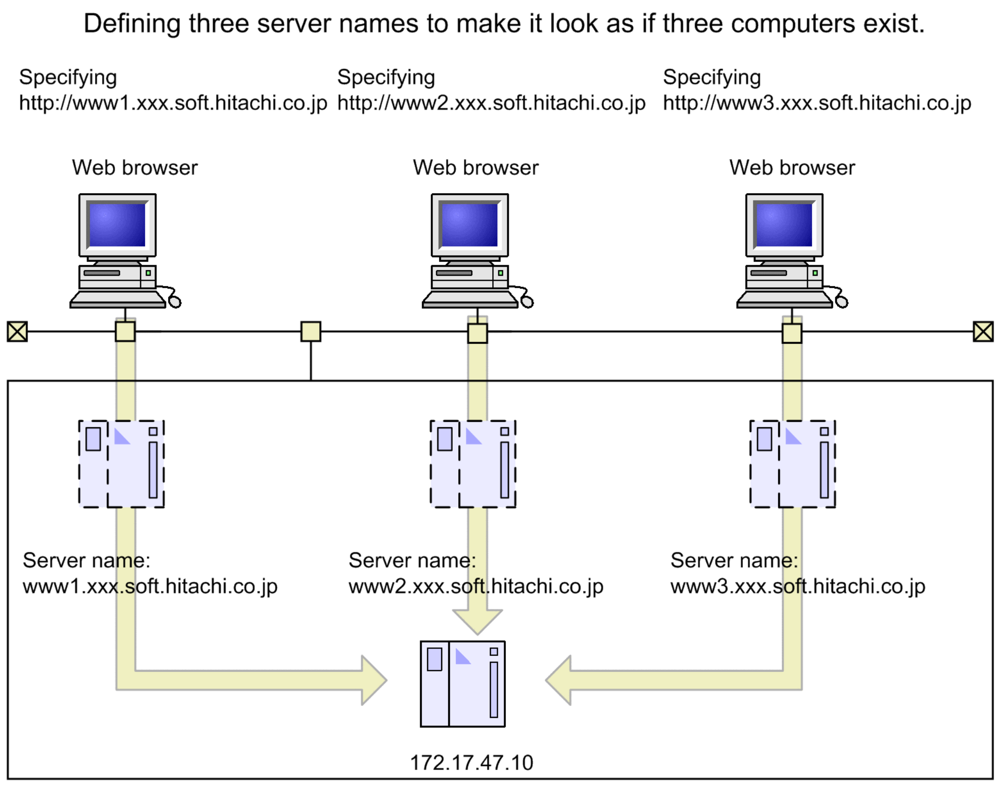
If the request from Web browser is http://www3.xxx.soft.hitachi.co.jp/, see <*Cosminexus-installation-directory*>*/httpsd/htdocs3/index.html* (when DirectoryIndex is specified as `index.html`).

However, you can use this method only when the host name (or the port number when required) is defined in the Host header during the request from the Web browser as Host: www1.xxx.soft.hitachi.co.jp. Note that this method cannot be used in old and simple Web browsers. In such cases, specifications of <VirtualHost> block coded at the topmost location are enabled (In this example www1.xxx.soft.hitachi.co.jp).

```
Port 80                                                         ... 1.
NameVirtualHost 172.17.40.10                                    ... 2.
<VirtualHost 172.17.40.10>                                      ... 3.
DocumentRoot "<Cosminexus-installation-directory>/httpsd/htdocs1"  ... 4.
ServerName www1.xxx.soft.hitachi.co.jp                          ... 5.
</VirtualHost>
<VirtualHost 172.17.40.10>                                      ... 6.
DocumentRoot "<Cosminexus-installation-directory>/httpsd/htdocs2"  ... 7.
ServerName www2.xxx.soft.hitachi.co.jp                          ... 8.
</VirtualHost>
<VirtualHost 172.17.40.10>                                      ... 9.
DocumentRoot "<Cosminexus-installation-directory>/httpsd/htdocs3"  ... 10.
ServerName www3.xxx.soft.hitachi.co.jp                          ... 11.
</VirtualHost>
```

1. A port number
2. IP address of the virtual host based on server name
3. Definition of virtual host 1
4. Definition of root directory
5. Definition of server name 1
6. Definition of virtual host 2
7. Definition of root directory
8. Definition of server name 2
9. Definition of virtual host 3
10. Definition of root directory
11. Definition of server name 3

#: You must register www1.xxx.soft.hitachi.co.jp, www2.xxx.soft.hitachi.co.jp, and www3.xxx.soft.hitachi.co.jp with the DNS server as host names of the host 172.17.40.10.


Defining three server names to make it look as if three computers exist.
Specifying http://www1.xxx.soft.hitachi.co.jp
Specifying http://www2.xxx.soft.hitachi.co.jp
Specifying http://www3.xxx.soft.hitachi.co.jp
Web browser
Web browser
Web browser
Server name: www1.xxx.soft.hitachi.co.jp
Server name: www2.xxx.soft.hitachi.co.jp
Server name: www3.xxx.soft.hitachi.co.jp
172.17.47.10

(2) *Virtual host based on IP address*

The IP address-based virtual host appears as multiple hosts to clients by the following three methods:

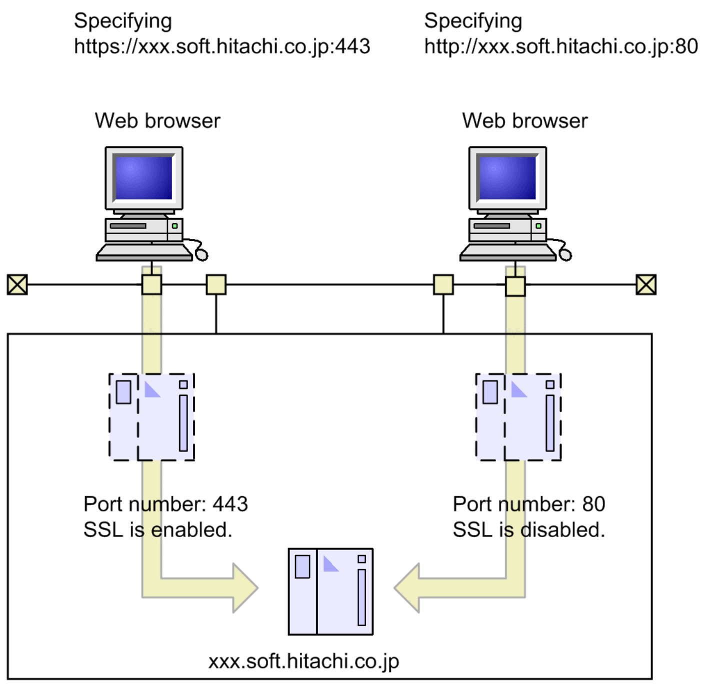
- Using multiple ports
- Specifying multiple network interfaces in a single server machine
- Specifying an alias of an IP address

Example 1: Open two ports on a Web server on a single server machine, and run as two hosts, one as a Web server that supports SSL and the other as a Web server that does not support SSL.

```
Listen 443                                                                                          ...1
Listen 80                                                                                           ...2
SSLDisable                                                                                          ...3
<VirtualHost xxx.soft.hitachi.co.jp:443>                                                            ...4
    DocumentRoot "<Cosminexus-installation-directory>/httpsd/ssldocs"
    SSLEnable                                                                                       ...5
    SSLCertificateFile "<Cosminexus-installation-directory>/httpsd/conf/ssl/server/
httpsd.pem"
    SSLCertificateKeyFile "<Cosminexus-installation-directory>/httpsd/conf/ssl/
server/httpsdkey.pem"
</VirtualHost>
<VirtualHost xxx.soft.hitachi.co.jp:80>                                                             ...6
    DocumentRoot "<Cosminexus-installation-directory>/httpsd/htdocs"
    SSLDisable                                                                                      ...7
</VirtualHost>
```

1. Definition of port number
2. Definition of port number
3. Disable SSL in the main server
4. Definition of virtual host of port number 443
5. Enable SSL
6. Definition of Virtual host of port number 80
7. Disable SSL

Defining two port numbers to make it look as if two computers exist.

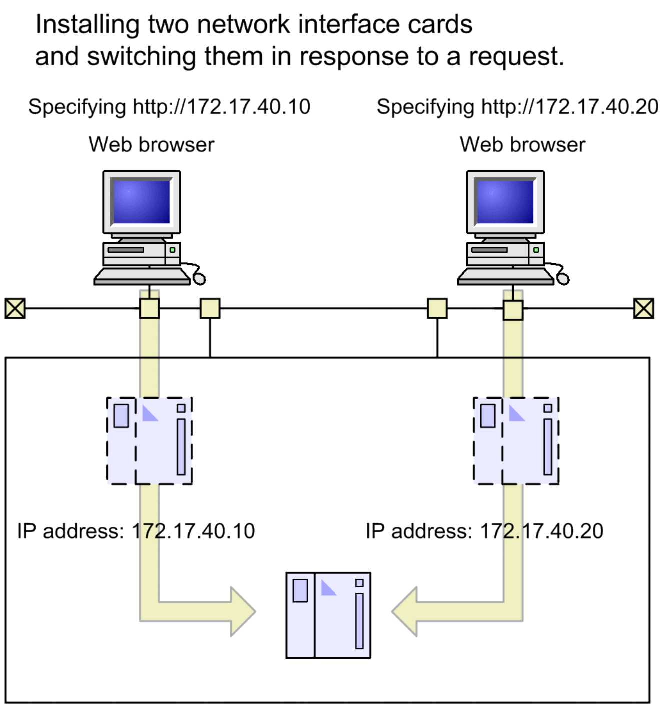


Example 2: Provide two NIC (Network Interface Card) (IP address: 172.17.40.10 and 172.17.40.20) on a single server machine, and switch hosts according to Web browser requests, on a single Web server.

If the request from Web browser is http://172.17.40.10/, see <*Cosminexus-installation-directory*>*/httpsd/htdocs1/index.html* (when DirectoryIndex is specified as index.html).

If the request from Web browser is http://172.17.40.20/, see <*Cosminexus-installation-directory*>*/httpsd/htdocs2/index.html* (when DirectoryIndex is specified as index.html).

```
Port 80
<VirtualHost 172.17.40.10>
DocumentRoot "<Cosminexus-installation-directory>/httpsd/htdocs1"
ServerName www10.xxx.soft.hitachi.co.jp
</VirtualHost>
<VirtualHost 172.17.40.20>
DocumentRoot "<Cosminexus-installation-directory>/httpsd/htdocs2"
ServerName www20.xxx.soft.hitachi.co.jp
</VirtualHost>
```


Installing two network interface cards
and switching them in response to a request.
Specifying http://172.17.40.10
Specifying http://172.17.40.20
Web browser
Web browser
IP address: 172.17.40.10
IP address: 172.17.40.20

4.4 Executing the CGI program on the Web server

CGI programs are programs that are executed on a Web server. You can use CGI programs for interactive Web access that is not possible when you access only static HTML.

(1) CGI program definition

There are three ways to execute CGI programs:

Specify a directory containing CGI programs in the ScriptAlias directive.

Specify the cgi-script handler in the file extension by using the AddHandler directive.

Specify the cgi-script handler in the SetHandler directive.

Hitachi recommends you to set the ScriptAlias directive in `httpsd.conf` for easy management of CGI programs.

(a) Example to specify the ScriptAlias directive

When the CGI program path is <*Cosminexus-installation-directory*>/httpsd/cgi-bin/ CGI program file name, and the clients access the /cgi-bin/CGI program file name

```
ScriptAlias /cgi-bin/ "<Cosminexus-installation-directory>/httpsd/cgi-bin/"
```

(b) Example to specify the AddHandler directive

- When .cgi is specified for the extension of files processed by the cgi-script handler

```
AddHandler cgi-script .cgi
```

Note that you need to specify the `ExecCGI` option in the Options directive.

(c) Example to specify the SetHandler directive

- When a cgi-script handler is specified for a request for the first file name in the script

```
<FilesMatch ^script>
  SetHandler cgi-script
  Options ExecCGI
</FilesMatch>
```

(2) Invoking CGI programs

Invoke CGI programs by specifying a URL from the Web browser in the following format:

```
http://host-name[:port-number]/path-name[?query-string]
```

- *host-name* [:*port-number*]

  Specify the host name or the IP address running the Web server, and the port number. If you omit the port number, the request will be sent to port number 80.

- *path-name*

  Specify the CGI program path for path name.

- *query-string*

  The *query-string* is a parameter to be passed to CGI programs. This parameter specifies a group of keywords and values. When you code data in the Web browser form, a query string is automatically set in the request line.

(3) Information to be passed to CGI programs

The Web server passes environment variables to CGI programs. For details, see *Appendix B Environment variables passed to CGI programs*.

(4) Example of CGI programs

This section describes the sample CGI programs and their execution examples.

Sample CGI programs

The following is an example of sample program source that can be used in Windows version. This program is written in the Perl language and the program name is test-cgi.pl:

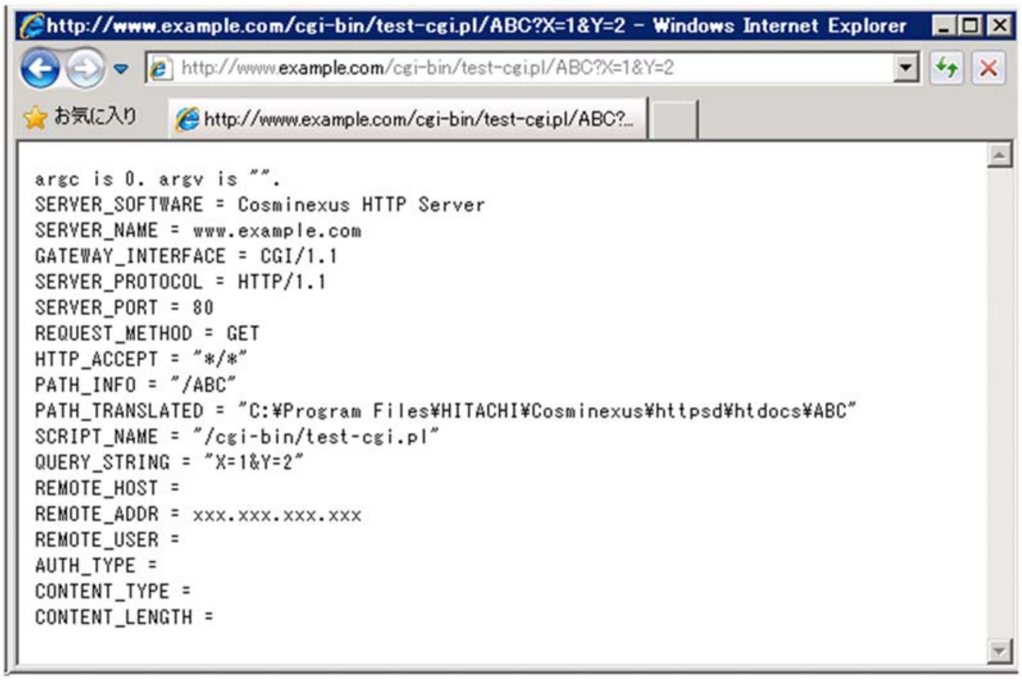
```
#! c:\bin\perl.exe

$argc=$#ARGV+1;
print "Content-Type: text/plain\n";
print "\n";
print "argc is $argc. argv is \"@ARGV\".\n";
print "SERVER_SOFTWARE = $ENV{'SERVER_SOFTWARE'}\n";
print "SERVER_NAME = $ENV{'SERVER_NAME'}\n";
print "GATEWAY_INTERFACE = $ENV{'GATEWAY_INTERFACE'}\n";
print "SERVER_PROTOCOL = $ENV{'SERVER_PROTOCOL'}\n";
print "SERVER_PORT = $ENV{'SERVER_PORT'}\n";
print "REQUEST_METHOD = $ENV{'REQUEST_METHOD'}\n";
print "HTTP_ACCEPT = \"$ENV{'HTTP_ACCEPT'}\"\n";
print "PATH_INFO = \"$ENV{'PATH_INFO'}\"\n";
print "PATH_TRANSLATED = \"$ENV{'PATH_TRANSLATED'}\"\n";
print "SCRIPT_NAME = \"$ENV{'SCRIPT_NAME'}\"\n";
print "QUERY_STRING = \"$ENV{'QUERY_STRING'}\"\n";
print "REMOTE_HOST = $ENV{'REMOTE_HOST'}\n";
print "REMOTE_ADDR = $ENV{'REMOTE_ADDR'}\n";
print "REMOTE_USER = $ENV{'REMOTE_USER'}\n";
print "AUTH_TYPE = $ENV{'AUTH_TYPE'}\n";
print "CONTENT_TYPE = $ENV{'CONTENT_TYPE'}\n";
print "CONTENT_LENGTH = $ENV{'CONTENT_LENGTH'}\n";
```

Executing CGI programs

Specify the following in the Web browser and invoke the sample CGI program:

```
http://www.example.com/cgi-bin/test-cgi.pl/ABC?X=1&Y=2
```

Result of sample program execution


http://www.example.com/cgi-bin/test-cgi.pl/ABC?X=1&Y=2 - Windows Internet Explorer
http://www.example.com/cgi-bin/test-cgi.pl/ABC?X=1&Y=2
お気に入り
http://www.example.com/cgi-bin/test-cgi.pl/ABC?...

```
argc is 0. argv is "".
SERVER_SOFTWARE = Cosminexus HTTP Server
SERVER_NAME = www.example.com
GATEWAY_INTERFACE = CGI/1.1
SERVER_PROTOCOL = HTTP/1.1
SERVER_PORT = 80
REQUEST_METHOD = GET
HTTP_ACCEPT = "*/*"
PATH_INFO = "/ABC"
PATH_TRANSLATED = "C:¥Program Files¥HITACHI¥Cosminexus¥httpsd¥htdocs¥ABC"
SCRIPT_NAME = "/cgi-bin/test-cgi.pl"
QUERY_STRING = "X=1&Y=2"
REMOTE_HOST =
REMOTE_ADDR = xxx.xxx.xxx.xxx
REMOTE_USER =
AUTH_TYPE =
CONTENT_TYPE =
CONTENT_LENGTH =
```

(5) Additional information to be passed to CGI programs

This section describes specification methods to pass information, other than the environment variables of CGI/1.1, to CGI programs on the Web server.

You can specify environment variables and their values to be passed to CGI programs in the configuration file. You can also specify environment variables that are not passed to CGI programs:

```
PassEnv environment variables                  specification of environment
variables to be passed to CGI programs
SetEnv environment variable values              specification of environment
variables and their values to be passed to CGI programs
UnsetEnv environment variables                 specification of environment
variables and their values that are not to be passed to CGI programs
```

(6) Defining environment variables

You can define environment variables as per client request. You can define environment variables and cancel the settings of environment variables using the host name and IP address of the client, who sends the request.

```
SetEnvIfNoCase Request_URI "\.(gif)|(jpg)$" request_is_image
```

In such cases, if the file extension is `.gif` or `.jpg` (this directive is not case sensitive), the environment variable *request_is_image* is passed to CGI programs.

(7) Points to be noted when using CGI programs on Windows

(a) Points to be noted when creating CGI programs

Standard input, standard output, and standard error output of CGI programs are used to send and receive data between CGI programs and server threads. The Timeout directive is enabled when data is sent and received. When you create CGI programs, after the sending and receiving of data finishes, either close or terminate functionality such as the standard input and output.

(b) Forced termination of CGI programs

When the Web server stops, a CGI program may not end until the process ends. For the forced termination of CGI programs, go to the 'Task manager' and end the program.

(8) Points to be noted when using CGI programs on UNIX

In CGI programs, the values specified in User and Group directives require execute permissions.

(9) Points to be noted when specifying path information

When the request URL contains the path information to be passed to a CGI program, the environment variable PATH_INFO is set to the path information and the environment variable PATH_TRANSLATED is set to a value that changes the path information into the path on the file system. When you change the path information into a path on the file system, the path specified in the DocumentRoot directive is the base path. For path information, if an optional name is specified in the Alias directive, change the path as per specifications.

According to Web server settings, if the path set in the environment variable PATH_TRANSLATED does not have access permissions, an access denied message is output to the error log. Even when the error message is output, the Web server runs the CGI program and continues the request processing. At this time, the environment variable PATH_TRANSLATED is also passed to the CGI program.

Example: The following is an example of error log output when the document root is <*Cosminexus-installation-directory*>/Cosminexus/httpsd, and the request from the Web browser is http://www.example.com/cgi-bin/test-cgi.pl/ABC (add "/ABC" as path information to the request that executes the CGI program "test-cgi.pl"):

```
[Fri Feb 20 12:00:00 2004] [error] [client 192.168.1.1] client denied by
server configuration: <Cosminexus-installation-directory>/Cosminexus/httpsd/
ABC
```

4.5 User authentication and access control

The methods to control the Web server access are as follows:

- Access control by user name and password
- Access control by host name or IP address of client
- Access control for directory
- Access control using directory service

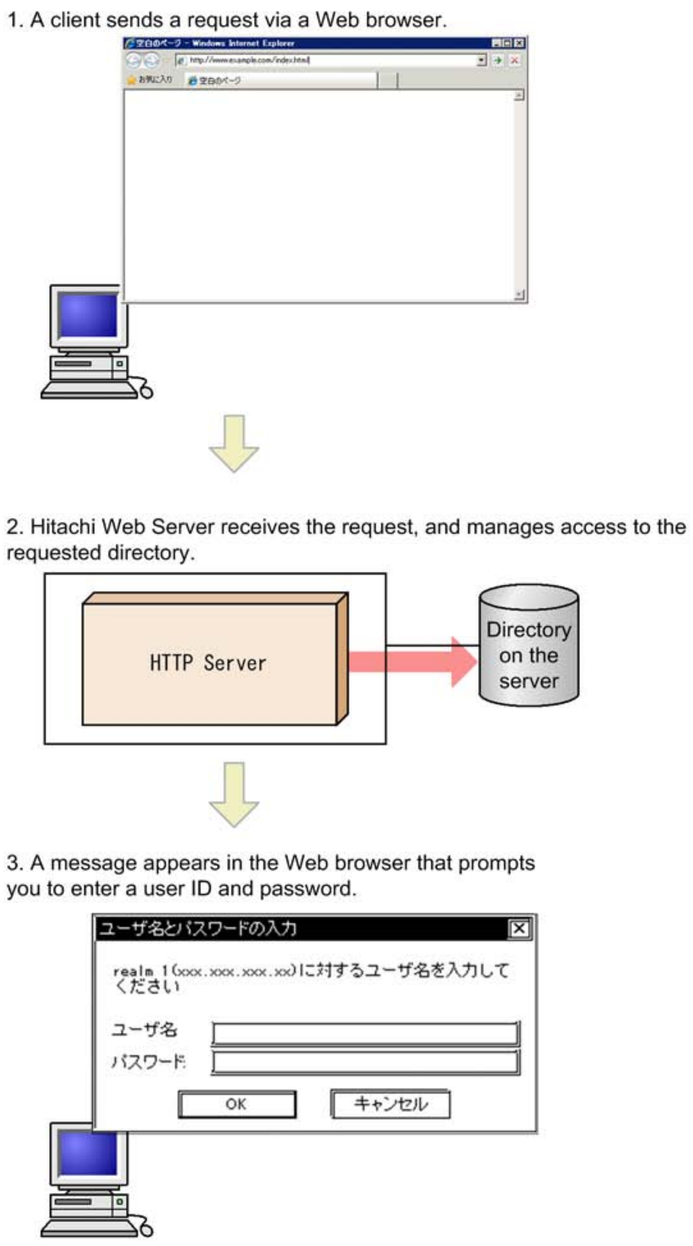
4.5.1 Access control by user name and password

Use the htpasswd command and register the user name and password in the password file. You can define access permissions to the directory and files in the host for a registered user name. For details on how to use the htpasswd command, see *(1) Registering the user name and password in the password file and changing the password.*

Example: The <*Cosminexus-installation-directory*>\httpsd\htdocs\directory is accessible only to specific users.

Use the htpasswd command and register the user name and password beforehand in the password file (<*Cosminexus-installation-directory*>\httpsd\htdocs\.htpasswd). Set the following directives in the `httpsd.conf` file. If a user accesses the <*Cosminexus-installation-directory*>\httpsd\htdocs\, the Web server responds with the status code 401 Authorization Required, and the Web browser requests for the user name and the password:

```
<Directory "<Cosminexus-installation-directory>/httpsd/htdocs">
    AuthType Basic
    AuthName "realm 1"
    AuthUserFile "<Cosminexus-installation-directory>/httpsd/htdocs/
.htpasswd"
    Require valid-user
</Directory>
```



1. A client sends a request via a Web browser.

2. Hitachi Web Server receives the request, and manages access to the requested directory.

3. A message appears in the Web browser that prompts you to enter a user ID and password.

realm 1(xxx.xxx.xxx.xx)に対するユーザ名を入力してください

空白のページ - Windows Internet Explorer
http://www.example.com/index.html
お気に入り
空白のページ
HTTP Server
Directory on the server
ユーザ名とパスワードの入力
ユーザ名
パスワード
OK
キャンセル

(1) Registering the user name and password in the password file and changing the password

You can register and change the user name and password in the password file using the htpasswd command.

How to use the htpasswd command is described below:

(a) Format

```
htpasswd [-b][-c | -D] password-file-name user-name[password]
```

(b) Parameters

- -b

  Specify this parameter when you specify the password in the command line.

- -c

  Specify this parameter when you create a new password file. You need not specify `-c` when you add a user and change the password in an already created password file.

- -D

  Specify this parameter when you delete a user registration. If the specified user is registered in the specified password file, the utility deletes the corresponding user from the password file.

- *password-file-name*

  Specify the password file that registers, changes, or deletes password.

- *user-name*

  Specify the user name for which password is to be registered, changed, or deleted.

- *password*

  Specify the password to be registered or changed. You can specify this parameter only when `-b` option is specified. .

(c) Usage method

If you specify the password file name, the user name to be registered, or the user name for which the password is to be changed, and run the htpasswd, the input of respective password is requested. If you enter the password twice, including the confirmation of password entry, the user name and the password of that user are registered in the password file:

```
 C:\>"<Cosminexus-installation-directory>\httpsd\bin\htpasswd.exe" .passwd
userxx     ...1.
 New password:                                                   ...2.
 Re-type new password:                                           ...3.
 Updating password for userxxx                                   ...4.
 C:\>
```

1. Change the password of userxx
2. Enter a new password
3. Re-enter the new password
4. End the registration of new password

When deleting the registration, start the htpasswd utility by specifying the `-D` option, the password file name, and the user name that is to be deleted.

```
 C:\>"<Cosminexus-installation-directory>\httpsd\bin\htpasswd.exe" -D
.passwd userxx  ...1.
 Deleting passwd for userxx                                      ...2.
 C:\>
```

1. Delete registration of userxx
2. Delete the registration of userxx and exit

(d) Note

- On Windows, the maximum size of a password is 128 characters. The maximum size of the user name is 128 characters. On UNIX, the maximum size of the password is the maximum length returned by the system call the getpass() (a read password functionality) or 128 characters, whichever value is smaller. For details on the getpass(), see the OS manual.
- When the htpasswd command is executed, a temporary work file is created in the directory in which the password file is created. The work file name is *password-file-name*`.`*process-ID*. The work file is deleted when the htpasswd command ends. However, the work file might not be deleted if you cancel to end the htpasswd utility while it is running. Manually delete the work file if it is not deleted automatically.

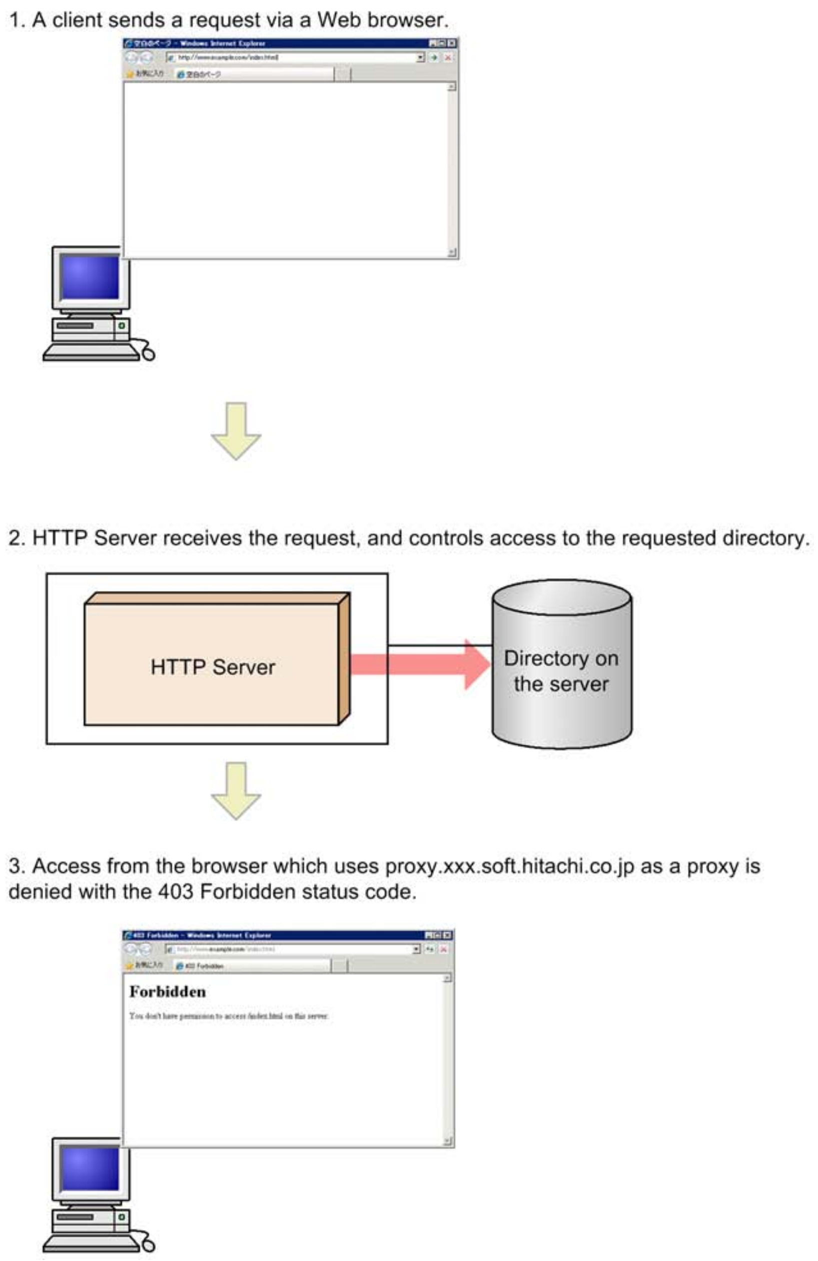
4.5.2 Access control by the host name or IP address of client

For the access control according to the host name and IP address of a client, use the directives Allow from and Deny from. Allow from directive specifies a host name that permits the access and Deny from directive specifies a host name that prohibits the access.

Example: Prohibit the requests received through the proxy to refer the directories under <*Cosminexus-installation-directory*>\httpsd\htdocs\directory.

Specify the following directives in the httpsd.conf file. When a user accesses <*Cosminexus-installation-directory*>\httpsd\htdocs\, the Web browser that uses the proxy proxy.xxx.soft.hitachi.co.jp, is denied access with the status code 403 Forbidden:

```
<Directory "<Cosminexus-installation-directory>/httpsd/htdocs">   Directory
definition
    Order deny,allow                        Define priority order of access
permission and prohibition
    Deny from proxy.xxx.soft.hitachi.co.jp  Prohibiting to access
</Directory>
```


1. A client sends a request via a Web browser.
2. HTTP Server receives the request, and controls access to the requested directory.
HTTP Server
Directory on the server
3. Access from the browser which uses proxy.xxx.soft.hitachi.co.jp as a proxy is denied with the 403 Forbidden status code.
403 Forbidden - Windows Internet Explorer
403 Forbidden

Forbidden

You don't have permission to access /index.html on this server.

4.5.3 Access control for directory

If you create an access control file (`.htaccess`) under a specific directory, you can set access permissions for that directory. Specify a client name (IP address) and a user name for which access is permitted or denied in the access control file.

(1) Access control file

If you create the access control file under a specific directory, you can set the access permissions for that directory. Specify the name of access control file in the AccessFileName directive. The default name is `.htaccess`.

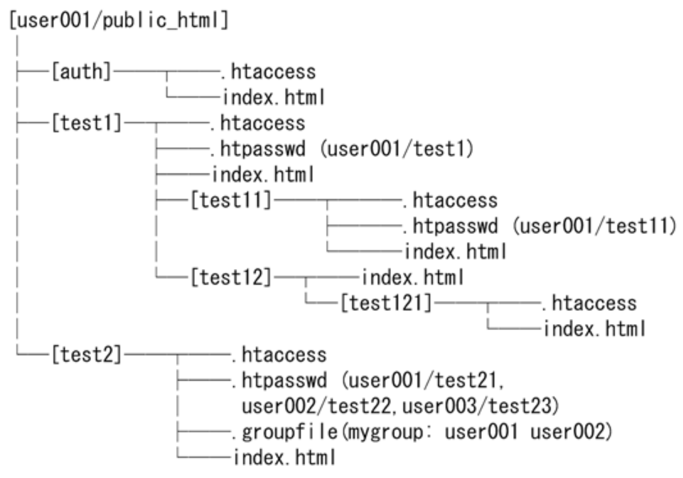
The access control from the access control file is enabled without restarting the Web server. However, for correct operations you need to set the AllowOverride directive of `httpsd.conf` to an appropriate level that allows overwriting.

If you specify a password file in the access control file, the server will request user to enter the user name and password when the user accesses the directory.

# The access control file (`.htaccess`) and password file (`.htpasswd`) need not have a one-to-one relationship. You can specify the same password file in the AuthUserFile directive for different access control files.

(2) Example to set the access permission

In a directory configuration such as the one given below, set the access permissions to access control files for each directory:



```
[user001/public_html]
 |
 ├──[auth]────┬───.htaccess
 |            └───index.html
 ├──[test1]──┬───.htaccess
 |           ├───.htpasswd (user001/test1)
 |           ├───index.html
 |           ├──[test11]────┬─────.htaccess
 |           |              ├─────.htpasswd (user001/test11)
 |           |              └─────index.html
 |           └──[test12]──┬────index.html
 |                        └──[test121]────┬───.htaccess
 |                                        └───index.html
 └──[test2]────┬───.htaccess
               ├───.htpasswd (user001/test21,
               |   user002/test22,user003/test23)
               ├───.groupfile(mygroup: user001 user002)
               └───index.html
```

■ *Defining access permissions under the auth directory (*`auth/.htaccess` *file)*

Access from the server with IP address 172.18.102.11 and 172.16.202.4 is denied:

```
Order deny,allow                      ...1.
Deny from 172.18.102.11 172.16.202.4  ...2.
```

1. First evaluate the access denial definition
2. Define the access denial

- *Defining access permissions under the test1 directory (*`test1/.htaccess` *file)*

  Allow access to test1/index.html and test1/test12/index.html only when the user enters the user name=user001 and password=test1.

```
AuthUserFile C:/user001/public_html/test1/.htpasswd      ...1.
AuthName "test1 Directory"                               ...2.
AuthType Basic
<Limit GET POST>                                         ...3.
   Require user user001                                  ...4.
</Limit>
```

1. Define the password file

   The user name and password registered in the password file

   User name: user001, Password: test1
2. Define the realm name
3. Define the method
4. Allow access to user name: user001

- *Defining access permissions under the* `test1/test11` *directory (*`test1/test11/.htaccess` *file)*

  Allow access to test1/test11/index.html only when the user enters user name=user001 and password=test11.

```
AuthUserFile C:/user001/public_html/test1/test11/.htpasswd      ...1.
AuthName "test11 Directory"                                     ...2.
AuthType Basic
<Limit GET POST>                                                ...3.
   Require user user001                                         ...4.
</Limit>
```

1. Define the password file

   The user name and password registered in the password file

   User name: user001, Password: test11
2. Define the realm name
3. Define the method
4. Allow access to user name: user001

- *Defining the access permissions under the* `test1/test12/test121` *directory (*`test1/test12/test121/.htaccess` *file)*

  Allow access to test1/test12/test121/index.html only when the user enters user name=user001, and password=test1, and the Web browser is MSIE.

```
Order deny,allow      ...1.
Allow from env=MSIE   ...2.
Deny from all         ...3.
```

1. First evaluate the access denial definition
2. Allow access if Web browser is MSIE
3. Deny access from all hosts

However, define the following directive in `httpsd.conf`:

```
SetEnvIf User-Agent ".*MSIE.*" MSIE
```

- *Defining access permissions under the* `test2` *directory (*`test2/.htaccess` *file)*

  Allow access to test2/index.html only when the user enters the user name and password of the mygroup group.

```
AuthUserFile C:/user001/public_html/test2/.htpasswd        ...1.
AuthGroupFile C:/user001/public_html/test2/.groupfile      ...2.
AuthName "test2 Directory"                                 ...3.
AuthType Basic
<Limit GET POST>                                           ...4.
   Require group mygroup                                   ...5.
</Limit>
```

1. Define the password file

   The user name and password registered in the password file

   User name: user001, password: test21

   User name: user002, password: test22

   User name: user003, password: test23

2. Define the group file

   The group name registered in the group file

Group name: mygroup

The user names registered in mygroup: user001, user002, user003

3. Define the realm name
4. Define the method
5. Allow access to the group name: mygroup

4.5.4 User authentication and access control using the directory service

You can link to the directory service (hereafter called the *LDAP server*), and can perform the user authentication without creating a password file. You can also control the access with the LDAP server attributes.

This functionality cannot be used for the HP-UX (IPF) version.

(1) Embedding the mod_hws_ldap module

The mod_hws_ldap module must be embedded for user authentication via the LDAP server. To embed the mod_hws_ldap module, specify the following directives in the configuration file (`httpsd.conf`):

- UNIX version

  LoadModule `hws_ldap libexec/mod_hws_ldap.so`

- Windows version

  LoadModule `hws_ldap modules/mod_hws_ldap.so`

(2) How to authenticate the access with LDAP server

When performing the user authentication, specify the AuthType directive and AuthName directive in a way similar to that of using a password file in <Directory>, `.htaccess`. By specifying the Require valid-user and the LDAPRequire directives, you can authenticate the user integrated with LDAP server.

The following is an example of user authentication when CGI of <*Cosminexus-installation-directory*>/httpsd/cgi-bin/ is used and the User Id and password are entered:

Example:

```
LDAPServerName ldap.server.hitachi.com
LDAPServerPort 389
<Directory "<Cosminexus-installation-directory>/httpsd/
cgi-bin">
AuthName LDAP-TASK
AuthType Basic
Require valid-user
```

```
    LDAPRequire
    </Directory>
```

You can also control the access with LDAP server attributes.

For example, you can also allow access only to users with the employee registration number from 100 to 200.

(3) Access control with LDAP server

You can define whether an authenticated user can use the relevant contents.

```
LDAPRequire [%DN attribute%] [LDAP-search-folder]
```

In *LDAP-search-folder*, the access permissions are defined on the basis of information registered in the LDAP server.

For example, among the authenticated users if you want to give access only to taro and hanako, define the access control as shown below. In advance register the defined information in the LDAP server.

Example:

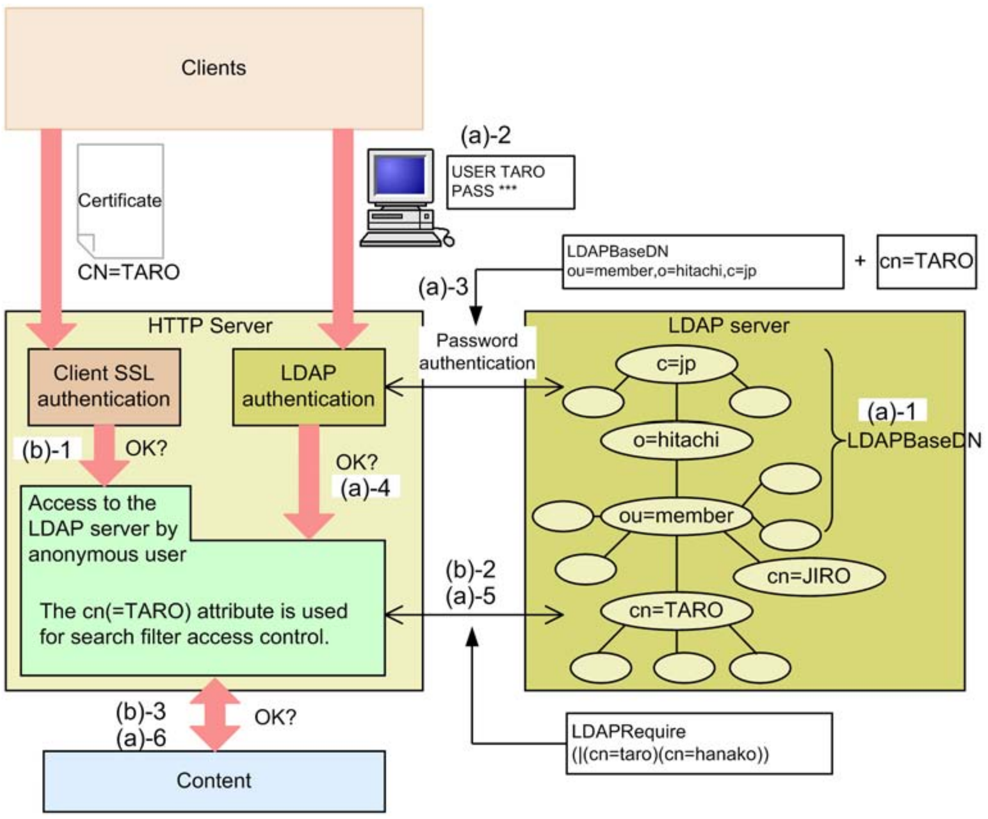
```
LDAPRequire %cn% (|(cn=taro)(cn=hanako))
```

(4) Relationship between the user authentication and access control

In the following example, user name is registered as `cn` in the LDAP server. When SSL client authentication and LDAPRequire directive are combined, Cosminexus HTTP Server checks whether the client is registered in the LDAP server after authentication depending upon the certificate. In such cases, Cosminexus HTTP Server treats the Common Name (CN) of subject in the client certificate as a user name, accesses the LDAP server as an anonymous access without using a password and checks. If Cosminexus HTTP Server cannot access the LDAP server, it responds with the status code 500 Internal Server Error.

The following figure shows the relationship between the user authentication and access control:

Figure 4-5: Relationship between the user authentication and access control



(a) Authentication by LDAP server

1. For the authentication by LDAP server, each user defines the registered DN (entry registered by the user to be authenticated: ou=member, o=Hitachi, and c=jp), in the LDAPBaseDN directive in advance.
2. When accessing the contents defined in the LDAPRequire directive, the Web server uses the information of DN and authenticates the client. A window requesting user name and password is displayed on the Web browser.
3. If a user name and password is entered, the Web server combines the cn=user name and the DN defined in the LDAPBaseDN directive, creates a user DN to be authenticated, and authenticates the password. In this case, the user DN will be cn=TARO, ou=member, o=hitachi, and c=jp.
4. If the password registered in the DN of LDAP server does not match with the password entered by the client, the Web server returns status code 401 Authorization Required and denies the access for the user.

5. Even if the passwords match, if there is a specification of LDAP search filter in the LDAPRequire directive, the Web server determines whether the search filter code matches with the user DN.
6. The Web server allows access if the code matches. Even if the password matches for cn=JIRO, since the search filter code does not match, the Web server responds with a status code (by default this is `401 Authorization Required`) according to the LDAPNoEntryStatus directive and denies access.

(b) When there is a SSL client certificate

1. When accessing from the client (Web browser), if the Web server receives an SSL client certificate, the LDAP server does not authenticate. SSL performs the authentication.
2. If LDAPRequire directive is defined, the Web server accesses the LDAP server and performs the access control.

   The Web server searches the LDAP server with the Common Name (CN) in the subject of the client certificate as the client name. If CN is does not exist in the LDAP server, the Web server responds with the status code 401 Authorization Required.
3. If CN of client certificate exist in the LDAP server, the Web server uses the search filter and confirms whether the CN of client certificate matches the filter. If the search filter is (|(cn=TARO)(cn=HANAKO)) and the CN certificate is TARO, the CN matches the search filter, so access is possible. If CN is JIRO, the CN does not match the search filter, so the Web server responds with the status code (by default this is `401 Authorization Required`) based on the LDAPNoEntryStatus directive and denies the access.

   LDAP server always possesses a `cn` to distinguish the user. Create the SSL client certificate based on the assumption that the certificate CN and this `cn` match with each other.

(5) User authentication by multiple LDAP servers

You can specify multiple LDAP servers in parallel. Accordingly, you can simultaneously use LDAP servers with different registered users and can authenticate these users. Note that you can change the LDAP server as per contents since directory-wise specification is also possible.

(a) Specifying multiple LDAP servers

You can specify the server name, port name and DN of multiple LDAP servers in the LDAPServerName, LDAPServerPort, and LDAPBaseDN directives. The LDAP server that was specified first has the highest priority, however the priority goes down in the specified order.

(b) Specifying the LDAP server for each directory

You can specify the following directives related to the LDAP for each directory. Inherit the specified directives from the top directory to the lower directories in the order of httpsd.conf, <VirtualHost>, and <Directory>.

LDAP related directives that can be specified in each directory

- LDAPServerName
- LDAPServerPort
- LDAPTimeout
- LDAPBaseDN

4.6 Displaying the file name list

The functionality that shows the list of file names in the directory on the Web browser is called the *Directory Index*. Define the following directives to enable the directory index functionality:

```
Options + Indexes
```

It is unsafe to display all the files from security point of view. In IndexIgnore directive, you need to specify files that are not to be displayed in the index.

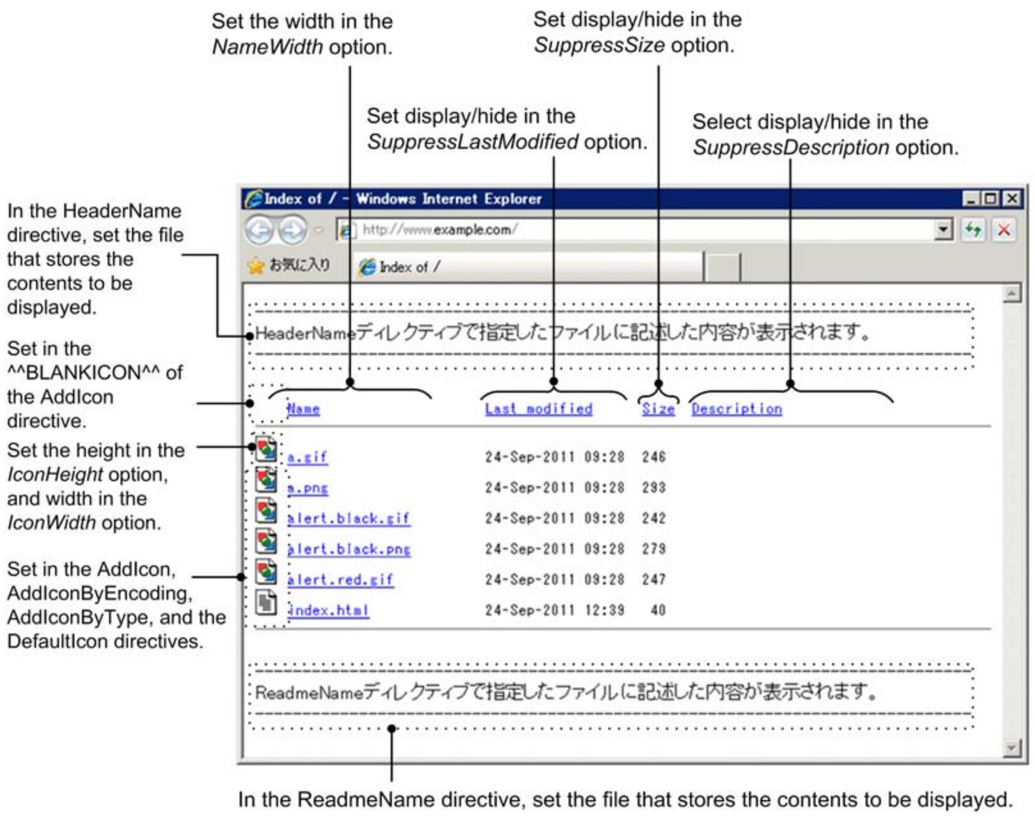
However, even if you specify Options + Indexes, when the files (by default `index.html`) specified in the DirectryIndex directive are under the directory, specified files are displayed.

When the directory index format is to be displayed, specify the following directive:

```
IndexOptions + FancyIndexing
```

Specify the detailed settings of the format display functionality in the IndexOptions directive and the AddIcon directive. The following figure shows the window displayed by the directory index functionality and the contents set in each directive:

Figure 4-6: Definition of the format display functionality


Set the width in the *NameWidth* option.
Set display/hide in the *SuppressSize* option.
Set display/hide in the *SuppressLastModified* option.
Select display/hide in the *SuppressDescription* option.
In the HeaderName directive, set the file that stores the contents to be displayed.
Set in the ^^BLANKICON^^ of the AddIcon directive.
Set the height in the *IconHeight* option, and width in the *IconWidth* option.
Set in the AddIcon, AddIconByEncoding, AddIconByType, and the DefaultIcon directives.
In the ReadmeName directive, set the file that stores the contents to be displayed.
Index of / - Windows Internet Explorer
http://www.example.com/
お気に入り
Index of /

HeaderNameディレクティブで指定したファイルに記述した内容が表示されます。

```
Name                   Last modified      Size  Description
a.gif                  24-Sep-2011 09:28  246
a.png                  24-Sep-2011 09:28  293
alert.black.gif        24-Sep-2011 09:28  242
alert.black.png        24-Sep-2011 09:28  279
alert.red.gif          24-Sep-2011 09:28  247
index.html             24-Sep-2011 12:39   40
```

ReadmeNameディレクティブで指定したファイルに記述した内容が表示されます。

Note that file names with multi-byte characters cannot be displayed.

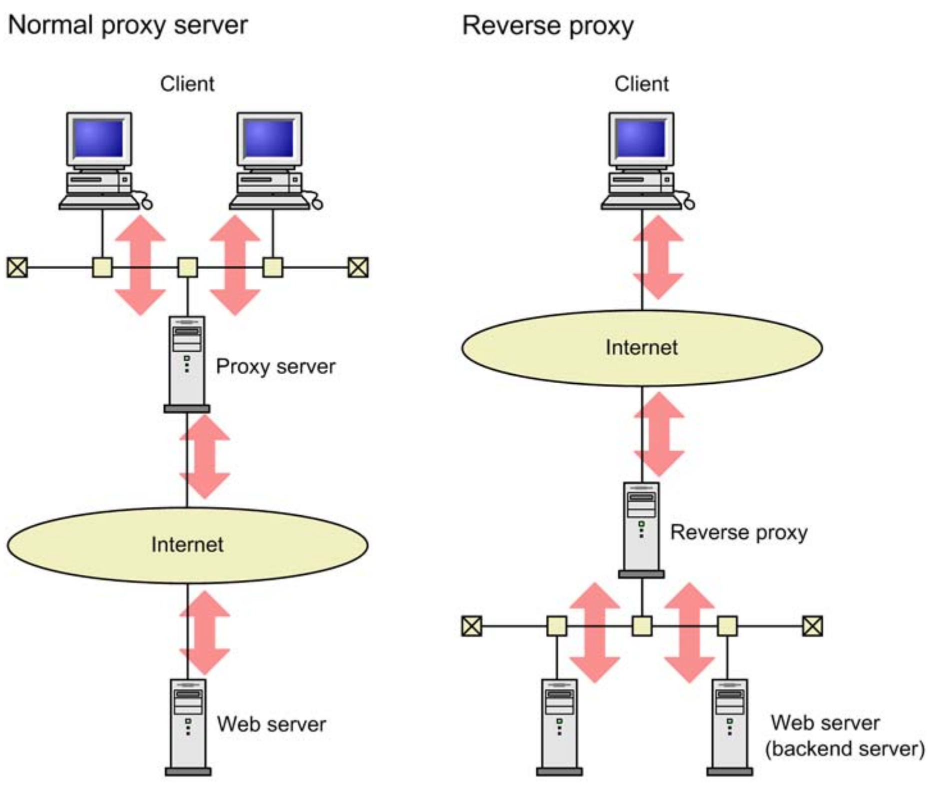
If the character sets used in the files specified in the HeaderName directive or the ReadmeName directive differ from the default character set (for the UNIX version: ISO-8859-1; for the Windows version: UTF-8), garbled text is displayed in the directory index. If this happens, specify the character sets used in the files specified in the HeaderName directive or ReadmeName directive by using the Charset option of the IndexOptions directive.

4.7 Setting the reverse proxy

Whenever a client cannot directly connect to the Internet, client requests are sent to the Web server via another server called a *proxy server*. The proxy server is usually installed at a connection point between the clients and the Internet. A proxy server that is installed at a connection point between the Internet and the Web server is called a *reverse proxy*. The reverse proxy processes requests received from clients on behalf of the Web server.

The following figure shows the difference between a normal proxy server and a reverse proxy.

Figure 4-7: Difference between a normal proxy server and a reverse proxy



You can use the reverse proxy for following operations:

- You can prevent direct access to contents.

  When important information is maintained in the Web server (such as database of credit card numbers), set the reverse proxy and the Web server on different machines, protect the Web server from invalid access and prevent the information leakage.

- You can integrate a highly loaded SSL process in the proxy server.

  If you use the reverse proxy and perform the SSL processing on another machine, you can distribute the load on the Web server.

- You can divide the load on the Web server without affecting the client.

  Even if you divide the Web server, since the proxy server acts as an alternative, the client can access with the same interface as prior to the division.

(1) Embedding proxy module

To use the reverse proxy, you need to embed a proxy module. To embed the proxy module, specify the following directives in the configuration file (`httpsd.conf`). For UNIX version, always specify the LoadModule directive in the following sequence:

- UNIX version

```
LoadModule proxy_module libexec/mod_proxy.so
LoadModule proxy_http_module libexec/mod_proxy_http.so
```

- Windows version

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so
```

(2) How to set directives

The following example describes the settings for each directive that sets the reverse proxy.

Each address is as follows:

Reverse proxy: www.example.com

Backend server: backend.example.com

(a) Reallocating the request URL and the request header

If you set the ProxyPass directive as shown below, the request http://www.example.com/news/oct-2001 received from client changes to the request http://backend.example.com/oct-2001:

```
ProxyPass /news/ http://backend.example.com/
```

The Host:Header is reallocated from "Host:www.example.com" to "Host:backend.example.com". After that, the reverse proxy sends the response from the backend server to the client.

(b) Reallocating the response header

If a Redirect directive is executed from a backend server and the directive uses an image map or contains a directory request that does not end with a backslash (/), the Location header in the backend server response will contain the backend server address. If the response is sent to the client as is, the client will request a redirect from the backend server directly, instead of from the reverse proxy. As a result, you must specify the ProxyPassReverse directive as follows, so that the redirect request also passes through the reverse proxy:

```
ProxyPassReverse /news/ http://backend.example.com/
```

The location header is changed to the reverse proxy address.

(c) Reassigning the Set-Cookie header

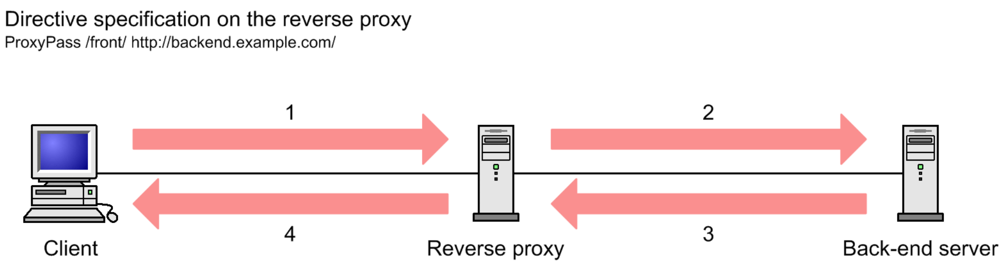
The domain name and path name are sometimes placed in a Set-Cookie header that is returned to the client from the backend server. By doing so, cookies are sent by the client only when the request matches the domain name and the path name in the Set-Cookie header.

Examples of when the Set-Cookie header is reassigned and when the Set-Cookie header is not reassigned are explained below.

■ Example of not reassigning the Set-Cookie header

The figure below illustrates an example of a reverse proxy sending a Set-Cookie header with a domain name and path name response from the backend server to the client as is. Note that the numbers in the following figure correspond to the explanation below.

Figure 4-8: Example of not reassigning the Set-Cookie header


Directive specification on the reverse proxy
ProxyPass /front/ http://backend.example.com/
1
2
3
4
Client
Reverse proxy
Back-end server

1: http://www.example.com/front/cgi-bin/test-cgi.pl
2: http://backend.example.com/cgi-bin/test-cgi.pl
3: Set-Cookie:～; domain=backend.example.com; path=/cgi-bin/
4: Set-Cookie:～; domain=backend.example.com; path=/cgi-bin/

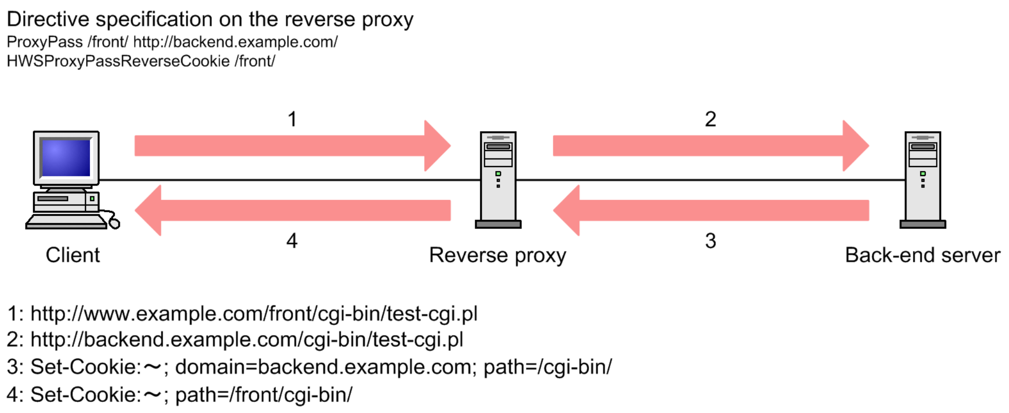
1. The client sends an http://www.example.com/front/cgi-bin/test-cgi.pl request to the reverse proxy.
2. The reverse proxy converts the URL, and then forwards it to the backend server.
3. The reverse proxy receives a Set-Cookie header from the backend server in which the domain name is set to domain=backend.example.com, and the path name is set to path=/cgi-bin/.
4. The reverse proxy returns the Set-Cookie header received from the backend server to the client as is.

In the above case, the client does not send the cookie received from the Set-Cookie header when the client sends a request to anything at or below /front/cgi-bin/ via the reverse proxy. This is because the domain name in the Set-Cookie header received by the client is backend.example.com, but the domain name of the reverse proxy is www.example.com. In the same way, the path names will also not match.

■ Example of reassigning the Set-Cookie header

The HWSProxyPassReverseCookie directive must be specified for a client to receive a cookie sent via the Set-Cookie header from the backend server. The figure below shows an example of reassigning the Set-Cookie header by specifying the HWSProxyPassReverseCookie directive. Note that the numbers in the following figure correspond to the explanation below.

Figure 4-9: Example of reassigning the Set-Cookie header


Directive specification on the reverse proxy
ProxyPass /front/ http://backend.example.com/
HWSProxyPassReverseCookie /front/
1
2
4
3
Client
Reverse proxy
Back-end server
1: http://www.example.com/front/cgi-bin/test-cgi.pl
2: http://backend.example.com/cgi-bin/test-cgi.pl
3: Set-Cookie:～; domain=backend.example.com; path=/cgi-bin/
4: Set-Cookie:～; path=/front/cgi-bin/

1. The client sends an http://www.example.com/front/cgi-bin/test-cgi.pl request to the reverse proxy.
2. The reverse proxy converts the URL, and then forwards it to the backend server.
3. The reverse proxy receives a Set-Cookie header from the backend server in which the domain name is set to domain=backend.example.com and the path name is set to path=/cgi-bin/.
4. The reverse proxy returns the reassigned Set-Cookie header to the client.

In the above case, the client receives a Set-Cookie header whose path name (/front/cgi-bin/) matches the beginning of the path in the request URL (/front/cgi-bin/test-cgi.pl). Also, there was no domain name in the Set-Cookie header received by the client. This is essentially the same as the domain name of the URL requested by the client (www.example.com) being specified in the Set-Cookie header. As such, cookies set by the Set-Cookie header can be sent with requests that go through reverse proxies to get to backend servers.

(3) Example of system building

This subsection shows examples of configuring a system by using Cosminexus HTTP Server on the reverse proxy and the backend server.

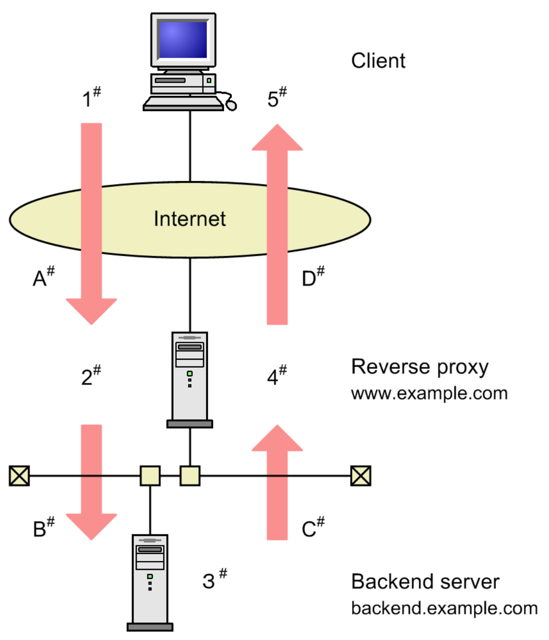
You must be aware of the redirect process and specify the appropriate settings when configuring the system. If the client accesses the URL of a directory on the backend server without adding a forward slash (/) at the end of the URL, the backend server sends a redirect request with the Location header. In this case, the Location header value must be changed from the backend server address to the reverse proxy address to ensure that all client re-requests go through the reverse proxy.

The system network configuration is shown in the following figure. Also, each address is as follows:

Reverse proxy: www.example.com

Backend server: backend.example.com

Figure 4-10: Network configuration



#: Corresponds to redirect processes (a) and (b) explained below.

(a) Recommended configuration

The host name and path name specified in the ProxyPass directive is the same as the host name and path name specified in the ProxyPassReverse directive. The ServerName directive is specified in all of the virtual hosts on the backend server, and the specification value is the same as the host name specified in the ProxyPassReverse directive on the reverse proxy.

The redirect process flow is shown in Table 4-6 when the reverse proxy and the backend server are configured as shown in Table 4-7 under the network configuration as illustrated in Figure 4-10.

Table 4-6: Example to set the recommended configuration

Setting location	Setting contents
Reverse proxy	ServerName www.example.com ProxyPass /before/ http://backend.example.com/after/ ProxyPassReverse /before/ http://backend.example.com/after/

Setting location	Setting contents
Backend server	ServerName backend.example.com

Table 4-7: Redirect process flow of recommended configuration

Location in the figure	Explanation
1	Access "http://www.example.com/before/dir".
2	Access "http://backend.example.com/after/dir" as per the ProxyPass directive value. Change and forward the Host header value in the backend.example.com.
3	Generate a URL that ends with a forward slash (/) because a forward slash (/) was not added to the end of the URL, set the URL in the Location header, and then return the redirect request.
4	Change and forward the Location header in the "http://www.example.com/before/dir/" as per the ProxyPassReverse directive value.
5	Access the "http://www.example.com/before/dir/" again as per the Location header.
A	Host header value is "www.example.com".
B	Host header value is "backend.example.com".
C	Location header value is "http://backend.example.com/after/dir/".
D	Location header value is "http://www.example.com/before/dir/".

#

When the backend server responds with a status code such as 302 Found or 404 Not found, the reverse proxy forwards the HTML document to the client without any changes. The backend server name mentioned in the HTML documents, such as 404 Not found and the redirect destination link address mentioned in 302 Found, are not changed to the reverse proxy information. Use the ErrorDocument directive on the backend server, or use the ProxyErrorOverride directive on the reverse proxy to prevent the client from seeing the backend server information.

(b) Configuration with the ProxyPreserveHost directive set to "On" in the reverse proxy

Normally, the reverse proxy converts the `Host` header value received from the client according to the ProxyPass directive value, and then forwards the converted value to the backend server. If you want to obtain the Host header value sent by the client as the Host header value in the backend server, set the value of the ProxyPreserveHost directive to On in the reverse proxy. Note the following points:

- Specify the same value as that of ServerName for the reverse proxy in the

ServerName directive on the backend server.

- Specify the same value for ServerName of the reverse proxy and for ServerName of the backend server for the host name of the ProxyPassReverse directive.

Table 4-8 shows the redirect process flow performed when the reverse proxy and the backend server are set as shown in Table 4-9 under the network configuration shown in the figure 4-10.

Table 4-8: Example configuration where the ProxyPreserveHost is set to `On` in the reverse proxy

Setting location	Setting contents
Reverse proxy	ServerName www.example.com ProxyPass /before/ http://backend.example.com/after/ ProxyPassReverse /before/ http://www.example.com/after/ ProxyPreserveHost On
Backend server	ServerName www.example.com

Table 4-9: Redirect process flow of configuration where the ProxyPreserveHost is set to `On` in the reverse proxy

Location in the figure	Explanation
1	Access "http://www.example.com/before/dir".
2	Access "http://backend.example.com/after/dir" as per the ProxyPass directive. As the ProxyPreserveHost directive is set to `On`, the Host header value will remain as www.example.com.
3	Generate a URL that ends with a forward slash (/), because a forward slash (/) was not added to the end of the URL, set the URL in the Location header, and then return the redirect request.
4	Change and forward the Location header in "http://www.example.com/before/dir/" as per the value of ProxyPassReverse directive.
5	Access the "http://www.example.com/before/dir/" again, as per the Location header.
A	The value of Host header is "www.example.com".
B	The value of Host header is "www.example.com".
C	The value of Location header is "http://www.example.com/after/dir/".
D	The value of Location header is "http://www.example.com/before/dir/".

(4) Note

(a) Basic points to be noted

- The reverse proxy sets the functions according to request URL patterns. So, you can specify settings so that the reverse proxy forwards specific requests to other backend servers as a reverse proxy and responds to other requests as a Web server. However, the settings make it unclear whether the requests are processed by the reverse proxy or by the Web server. Therefore, when you use a reverse proxy, we recommend that you specify the following settings to forward all the requests from the reverse proxy to the backend server.

```
ProxyPass / http://forwarding-destination-backend-server-address/
```

  If you use a reverse proxy and a Web server together, you can separate the functionality amongst the virtual hosts.

- The reverse proxy stores the Host header value received from the client into the X-Forwarded-Host header, converts the Host header value to the specification value of the ProxyPass directive, and then forwards the converted value to the backend server. If you want to refer to the Host header value sent by the client on the backend server, refer to the X-Forwarded-Host header value sent by the reverse proxy. However, if the ProxyPreserveHost directive is set to On, directly refer to the Host header sent by the reverse proxy.
- When you access the backend server via the reverse proxy, you must specify not the URL of the backend server, but the URL to which the reverse proxy accesses the HTML content provided by the backend server. When you enter the reference URL for content such as images and style sheets, the same care must be taken.

  Example:

  Set up the link from the index.html to index2.html in the following states:

  - On the reverse proxy, the ProxyPass directive value is specified as /before/ http://*backend-server-address*/after/.
  - On the backend server, both index.html and index2.html exist in the same directory (/after/ below).

  The following table shows the relationship between the coding method and accessibility of index.html:

Table 4-10: Relationship between the coding method and accessibility of link

Coding of link	Accessibility when link is clicked
<A HREF="index2.html">link</A>	Y

Coding of link	Accessibility when link is clicked
<A HREF="/before/index2.html">link</A>	Y
<A HREF="http://reverse proxy address/before/index2.html">link</A>	Y
<A HREF="/after/index2.html">link</A>	N

- Reverse proxy does not support HTTP version 0.9.

(b) Points to be noted for the ProxyPass directive

- When the path name specified in the ProxyPass directive and the request URL are same, or when the path name is included from the beginning of request URL, the path name is determined as matching.

  However, when there is no /(slash) at the end of path name and if the path name matches exactly with the request URL, or the path name is included from the beginning as the directory, the path name is determined as matching.

  If the path name is matching, delete the beginning part of URL that is similar to the path name and add the remaining part to the path name that is specified in the ProxyPass directive, and then send the request.

  In the ProxyPass directive, specify a path name that ends with /(slash). The following table describes the relationship between specifications of ProxyPass directive and the request:

  Table 4-11: Relationship between specifications of ProxyPass directive and request

Example of ProxyPass directive specification	Request	Match	Location where request is forwarded
ProxyPass /abc/ http://backend.example.com/	http://reverse proxy address/abc/	Y	http://backend.example.com/
	http://reverse proxy address/abc	N	--
	http://reverse proxy address/abc/def	Y	http://backend.example.com/def
ProxyPass /abc http://backend.example.com/	http:// reverse proxy address /abc	Y	http://backend.example.com/
	http:// reverse proxy address /abc/	Y	http://backend.example.com//

Example of ProxyPass directive specification	Request	Match	Location where request is forwarded
	http:// reverse proxy address /abc/def	Y	http://backend.example.com//def

Legend:

Y: Match.

N: Does not match.

--: Not applicable.

- If you specify multiple ProxyPass directives and the requested URL matches with multiple path names, the ProxyPass directive that is specified first is applied.

  Example:

  Backend server that processes requests for /abc/def/: backend1.example.com

  Backend server that processes requests for /abc/ other than to /abc/def/: backend2.example.com

  Backend server that processes all other requests: backend3.example.com

  Specify these settings in the following sequence:

```
ProxyPass /abc/def/ http://backend1.example.com/
ProxyPass /abc/ http://backend2.example.com/
ProxyPass / http://backend3.example.com/
```

- The request URL specified in the ProxyPass directive is forwarded before performing the corresponding file search in the local process, which is a feature of the Web server. Therefore, when the request URL matches the path name specified in the ProxyPass directive, even if a file matching the request URL exists, the request URL is converted into a backend server request, and then forwarded to the backend server.
- If the specified directory in a request URL is not closed by a / (slash), the reverse proxy does not respond with the redirect.

  Example: In the case of ProxyPass /ab/ http://backend.example.com/

  If a request of http://reverse proxy is address/ab, the request is doe not match, and if /ab is not available in the reverse proxy, the Web server responds with '404 Not Found'.

(c) Notes for the ProxyPassReverse directive

- If the URL specified in the ProxyPassReverse directive and the Location header value received from the backend server are exactly same, or if the URL includes the prefix of the request URL, the URL specified in the ProxyPassReverse directive and the Location header value are seen as matching. If they match, the address is sent to the client as the reverse proxy according to the specification of the ProxyPassReverse directive.

  (Example) When the Location header sent as a response by the backend server is Location: http://backend-server-address/docs/memo/

  > If the ProxyPassReverse directive is set to ProxyPassReverse /path/ http://*backend-server-address*/docs/, the Location header to be returned to the client is Location: http://*reverse-proxy-address*/path/memo/.

  If you specify multiple ProxyPassReverse directives, the directive that is specified first is applied.

- When the reverse proxy changes the value of Location header according to the value set in ProxyPassReverse directive and forwards it to the client, set the value used in the current connection for the Location header scheme. For example, http is set when accessing by http. Therefore, when a request is redirected to https by using the Location header at the time of access via http, set the host name of the reverse proxy to the Location header value on the backend server to prevent the Location header from matching the value of the ProxyPassReverse directive.

(d) Notes for the HWSProxyPassReverseCookie directive

- The HWSProxyPassReverseCookie directive is specified to convert the Set-Cookie header sent by the backend server as a response. By setting the same value as the path name of the ProxyPass directive for the HWSProxyPassReverseCookie directive, the Set-Cookie header is converted for each ProxyPass directive.
- The table below explains the conversion rules of the Set-Cookie header when the directives of the reverse proxy are specified as follows:

```
ProxyPass /front/ http://backend.example.com/
HWSProxyPassReverseCookie /front/
```

Table 4-12: Conversion rules of the Set-Cookie header

#	Set-Cookie header sent to the client as a response	Set-Cookie header sent by the backend server as a response	Explanation for the conversion rule
1	Set-Cookie: ~; path=/front/	Set-Cookie: ~; path=/	When the domain name is not specified in the Set-Cookie header received from the backend server, the reverse proxy replaces the forward slash (/) of the path name in the Set-Cookie header with /front/.
2	Set-Cookie: ~; path=/front/	Set-Cookie: ~; domain=backend.example.com; path=/	When the domain name in the Set-Cookie header received from the backend server matches the domain name of the forwarding destination URL specified in the ProxyPass directive exactly, the reverse proxy replaces the forward slash (/) of the path name in the Set-Cookie header to /front/. The reverse proxy deletes the domain name in the Set-Cookie header, and then sends the Set-Cookie header to the client.
3	Set-Cookie: ~; domain=.example.com; path=/	Set-Cookie: ~; domain=.example.com; path=/	When the domain name in the Set-Cookie header received from the backend server starts with a period (.), the reverse proxy sends the Set-Cookie header received from the backend server to the client without any change.
4	Set-Cookie: ~; domain=other.example.com; path=/	Set-Cookie: ~; domain=other.example.com; path=/	When the domain name in the Set-Cookie header received from the backend server differs from the domain name of the forwarding destination URL specified in the ProxyPass directive, the reverse proxy sends the Set-Cookie header received from the backend server to the client without any changes.

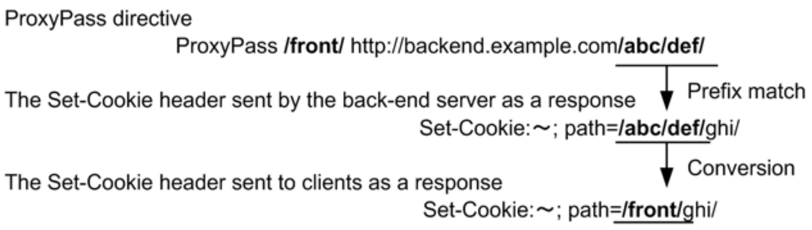
#	Set-Cookie header sent to the client as a response	Set-Cookie header sent by the backend server as a response	Explanation for the conversion rule
5	Set-Cookie: ~	Set-Cookie: ~	When the domain name and the path name are not specified in the Set-Cookie header, the reverse proxy sends the Set-Cookie header received from the backend server without any changes.

- Here are explanations for the conversion rules of the Set-Cookie header when the directives of the reverse proxy are specified as follows:

```
ProxyPass /front/ http://backend.example.com/abc/def/
HWSProxyPassReverseCookie /front/
```

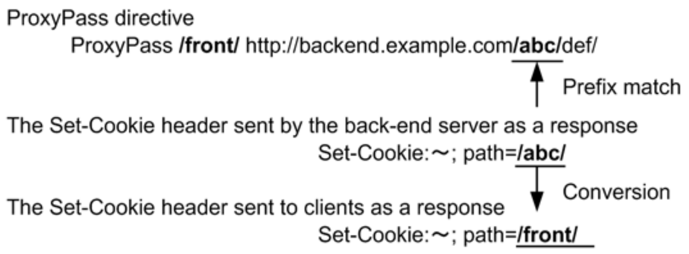
- When the path name sent by the backend server is /abc/def/ghi/

  When the path name of the forwarding destination URL specified in the ProxyPass directive matches the prefix of the path name set in the Set-Cookie header, the matched part of the path name in the Set-Cookie header is replaced to the path name specified in the ProxyPass directive.


ProxyPass directive
ProxyPass /front/ http://backend.example.com/abc/def/
Prefix match
The Set-Cookie header sent by the back-end server as a response
Set-Cookie:～; path=/abc/def/ghi/
Conversion
The Set-Cookie header sent to clients as a response
Set-Cookie:～; path=/front/ghi/

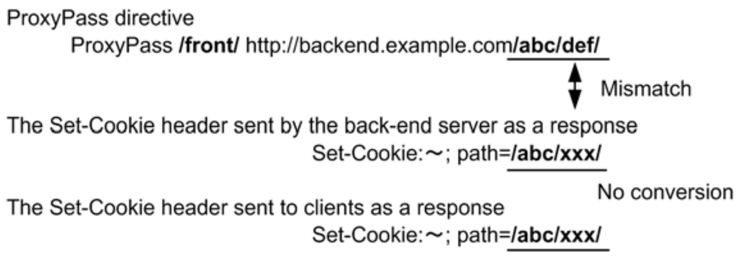
- The path name of the Set-Cookie header sent by the backend server is /abc/

  When the path name set in the Set-Cookie header matches the prefix of the path name of the forwarding destination URL specified in the ProxyPass directive, the path name in the Set-Cookie header is replaced to the path name in the ProxyPass directive.


ProxyPass directive
ProxyPass /front/ http://backend.example.com/abc/def/
Prefix match
The Set-Cookie header sent by the back-end server as a response
Set-Cookie:～; path=/abc/
Conversion
The Set-Cookie header sent to clients as a response
Set-Cookie:～; path=/front/

- The path name of the Set-Cookie header sent by the backend server is /abc/xxx/

  When the path name of the forwarding destination URL specified in the ProxyPass directive does not match the path name set in the Set-Cookie header, the reverse proxy does not perform the Set-Cookie header conversion. The reverse proxy sends the Set-Cookie header received from the backend server to the client without any change.


ProxyPass directive
ProxyPass /front/ http://backend.example.com/abc/def/
Mismatch
The Set-Cookie header sent by the back-end server as a response
Set-Cookie:～; path=/abc/xxx/
No conversion
The Set-Cookie header sent to clients as a response
Set-Cookie:～; path=/abc/xxx/

(e) Points to be noted for performance

When the document name or the host name is specified in the ProxyPass directive, a DNS query is generated. If you already know the IP address of the backend server, you can reduce the time for resolving the name by mentioning the IP address in the hosts file in advance.

4.8 Displaying the operation status (Status information display)

Display number of running processes, number of standby processes, and the status of each process (such as R, W, and L) in the Web browser (number of server threads for Windows version). On the basis of this information, you can tune the StartServers, MinSpareServers, MaxSpareServers, and the MaxClients directive (the ThreadsPerChild directive for Windows version). For details on each directive, see *4.1 Relationship between processes and directives of Cosminexus HTTP Server*.

When you set the ExtendedStatus directive to `On`, more detailed information is displayed.

(1) Specifying the server-status handler

Specify the server-status handler as follows to use the display functionality of status information:

```
<Location /server-status>
      SetHandler server-status
</Location>
```

However, generally the access to the status information of the Web server is controlled and the information is not disclosed to the end-user.

(2) Specifying URL

To display the status information, specify the URL from Web browser in the following format. Note that the server status may be displayed incorrect temporarily depending upon the timings:

```
http://Host-name[:Port-number]/server-status[?{refresh=Update
interval|auto|notable}]
```

You can specify refresh=updated interval, auto, and notable respectively with the conjunction &. However, the auto is in the plain text, and hence there is no significance of specifying auto together with notable.

- refresh=update interval ((1-3600))

  Specify the interval in which the Web server status information is to be updated, in seconds. However, the Web browser needs the functionality to support the Refresh header of HTTP response header. If a value not within the acceptable range is specified, 60 seconds is set.

- auto

  This is displayed in the plain text format. As the `auto` is displayed in the plain text, you can easily use `auto` in other programs.

- notable

  Display the status information in HTML without using the <TABLE> tag.

example-of-specification

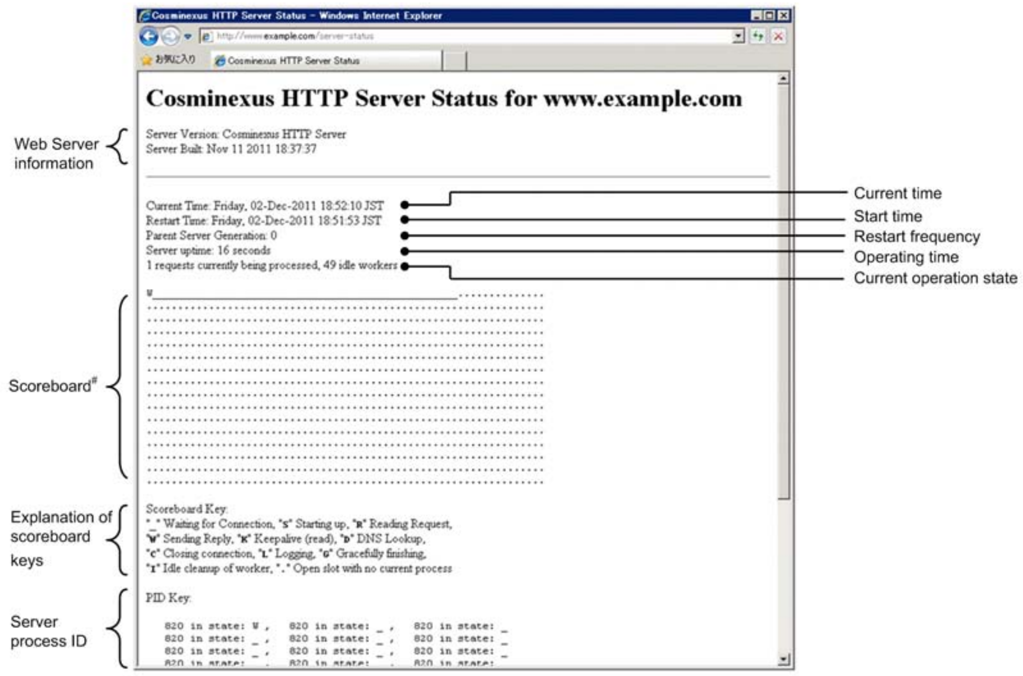
```
http://www.example.com/server-status?refresh=60&notable
```

example-of-display

```
http://www.example.com/server-status
```

The following figure shows the example of status information display, when the URL is specified as above. The display format varies slightly for UNIX version and Windows version.

Figure 4-11: Example of status information display


Cosminexus HTTP Server Status - Windows Internet Explorer
http://www.example.com/server-status
お気に入り
Cosminexus HTTP Server Status
Web Server information
Scoreboard#
Explanation of scoreboard keys
Server process ID
Current time
Start time
Restart frequency
Operating time
Current operation state

Cosminexus HTTP Server Status for www.example.com

Server Version: Cosminexus HTTP Server
Server Built: Nov 11 2011 18:37:37

Current Time: Friday, 02-Dec-2011 18:52:10 JST
Restart Time: Friday, 02-Dec-2011 18:51:53 JST
Parent Server Generation: 0
Server uptime: 16 seconds
1 requests currently being processed, 49 idle workers

```
W_________________________________________________..............
................................................................
................................................................
................................................................
................................................................
................................................................
................................................................
................................................................
................................................................
................................................................
................................................................
................................................................
................................................................
................................................................
................................................................
................................................................
```

Scoreboard Key:
"_" Waiting for Connection, "S" Starting up, "R" Reading Request,
"W" Sending Reply, "K" Keepalive (read), "D" DNS Lookup,
"C" Closing connection, "L" Logging, "G" Gracefully finishing,
"I" Idle cleanup of worker, "." Open slot with no current process

PID Key:

```
   820 in state: W ,    820 in state: _ ,    820 in state: _
   820 in state: _ ,    820 in state: _ ,    820 in state: _
   820 in state: _ ,    820 in state: _ ,    820 in state: _
   820 in state: _ ,    820 in state: _ ,    820 in state: _
```

# In the scoreboard, the state of running server processes is shown with keys.

Legend:

_	··· Waiting for a request
S	··· Starting up
R	··· Receiving a request from a client
W	··· Processing the request, and sending the response to a client
K	··· Waiting to receive a request in the KeepAlive state
D	··· Lookup in-progress (see the HostnameLookups directive)
C	··· Closing connection
L	··· Outputting logs
G	··· Waiting to finish the graceful restart process
I	··· Terminating threads
.	··· Not running

(3) Information that can be acquired

The following table describes the information that you can acquire by the status information display functionality. You can acquire the detailed information by setting the ExtendedStatus directive to `On`.

Table 4-13: Information which can be acquired by using the status information display functionality (without the auto specification)

No.	Contents	Explanation	Value and acquisition of ExtendedStatus	
			Off	On
1	Server Version	Server version	Y	Y
2	Server Built	Server building time	Y	Y
3	Current Time	Current time	Y	Y
4	Restart Time	Start time	Y	Y
5	Parent Server Generation	Number of times the server process is restarted (initial value 0)	Y	Y
6	Server uptime	Server process running time	Y	Y
7	Total accesses	Number of total accesses	N	Y
8	Total Traffic	Total amount of traffic	N	Y
9	CPU Usage: u vvv s www cu xxx cs yyy - zzz% CPU load	User time, system time, user time of the child process, system time of the child process, CPU usage rate (UNIX version)	N	Y
10	xxx requests/sec - yyy B/second - zzz B/request	Number of requests per second, traffic per second, traffic per request	N	Y
11	xxx requests currently being processed, yyy idle workers	The number of server processes during request processing (threads), the number of server processes in the status of request waiting (threads)	Y	Y
12	Scoreboard	Operation status of a single thread	Y	Y
13	Scoreboard Key	Legend of scoreboard	Y	Y
14	PID Key	Server process ID and the operation status of a single thread	Y	Y
15	Srv	Identifiers and restart frequency of server process	N	Y
16	PID	Process ID	N	Y
17	Acc	Access count (connection-wise/ thread-wise/ slot-wise)	N	Y

No.	Contents	Explanation	Value and acquisition of ExtendedStatus	
			Off	On
18	M	Operation status	N	Y
19	CPU	CPU time (in seconds) (UNIX version)	N	Y
20	SS	Elapsed seconds from the starting of last process	N	Y
21	Req	Milliseconds required for the last process	N	Y
22	Conn	Traffic for the connection	N	Y
23	Child	Traffic of process	N	Y
24	Slot	Traffic of slot	N	Y
25	Client	Client of last process	N	Y
26	VHost	Virtual host name	N	Y
27	Request	Request line of last process	N	Y

Legend:

Y: Can be acquired.

N: Cannot be acquired.

Table 4-14: Information which can be acquired by using the status information display functionality (with the auto specification)

No	Contents	Explanation	Value and acquisition of `ExtendedStatus`	
			Off	On
1	Total accesses	Total number of accesses	N	Y
2	Total kBytes	Total amount of traffic	N	Y
3	CPULoad	CPU usage rate (UNIX version)	N	Y
4	Uptime	Server process running time (in seconds)	N	Y

No	Contents	Explanation	Value and acquisition of `ExtendedStatus`	
			Off	On
5	ReqPerSec	The number of requests per second	N	Y
6	BytesPerSec	Amount of traffic per second	N	Y
7	BytesPerReq	Amount of traffic per request	N	Y
8	BusyWorkers	The number of server processes during request processing (threads)	Y	Y
9	IdleWorkers	The number of server processes in the status of request waiting (threads)	Y	Y
10	Scoreboard	Operation status of each thread	Y	Y

Legend:

Y: Can be acquired.

N: Cannot be acquired.

(4) Note

Multibyte characters might be set in the information for the time zones of **Current Time** and **Restart Time** displayed by using the server status display functionality. In this case, Cosminexus HTTP Server escapes all of the character strings (replaces them with the character strings configured with the prefixes which start with \x and hexadecimal codes).

4.9 Flow-restricting functionality

When the Web server load increases due to an increase in Web server access or impact of a business application, a functionality called *flow-restricting functionality* maintains the efficiency of Web service process by restricting the number of users who access the Web site.

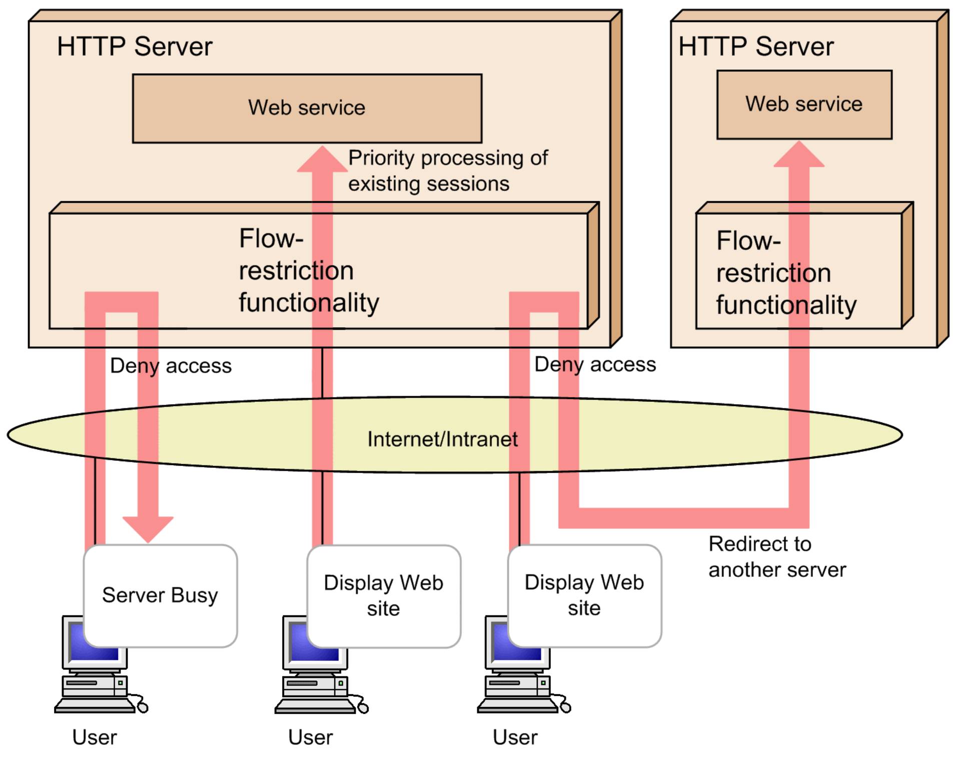
You can use flow-restricting functionality by embedding the mod_hws_qos module in Cosminexus HTTP Server. You can achieve the following by using the flow-restricting functionality. Hereafter, 'number of server processes' refers to the 'number of server threads' for Windows version.

- You can restrict the number of users who access to the Web site at the same time if you restrict the number of server processes that execute the request processes. When there is a high load, the server immediately returns a denial response for the request that exceeds the restricted value and redirects the request to another server. In this way you can maintain the response time.
- You can use session management with cookies to deny a new request when the server is highly loaded and can maintain the response time for the users who are already accessing the server.

For providing an adequate amount of Web service, you need a sufficient number of server processes that can satisfy the number of users accessing the Web service at the same time. If one Web server is not enough, arrange for multiple Web servers, distribute the access by load distribution functionality, and design the operation for assured Web services.

As mentioned above, prepare the Web service resources, and also use the flow-restricting functionality to prepare for the temporary overloaded status. The following figure shows an overview of flow-restricting functionality using mod_hws_qos:

Figure 4-12: Overview of the flow-restricting functionality using mod_hws_qos



(1) Embedding the mod_hws_qos module

You need to embed the mod_hws_qos module to use the flow-restricting functionality. Specify the following directives in the configuration file (`httpsd.conf`) to embed the mod_hws_qos module:

- UNIX version

```
LoadModule hws_qos libexec/mod_hws_qos.so
```

- Windows version

```
LoadModule hws_qos modules/mod_hws_qos.so
```

(2) How to set directives

Following examples describe the settings of each directive for the flow-restricting functionality:

(a) Request denial depending upon the restriction on number of server processes

If the specifications are as follows, and number of server threads being requested is 13, a new request is denied with the status code 503.

- UNIX version

```
MaxClients 15
QOSRejectionServers 2
QOSCookieServers 0
```

- Windows version

```
ThreadsPerChild 15
QOSRejectionServers 2
QOSCookieServers 0
```

(b) Session management using cookies

The session management performed by using cookies provides the HWS creation mode, in which the cookies generated on Cosminexus HTTP Server are used, and the user creation mode, in which the cookies generated on external modules other than on Cosminexus HTTP Server are used. You can select which mode is to be used by using the QOSCookieName directive.

HWS creation mode

When a request process is executed, the cookie generated on Cosminexus HTTP Server is added to the Set-Cookie response header. Requests with a cookie generated by Cosminexus HTTP Server are processed with a higher priority than requests without cookies.

User creation mode

When a cookie which is not generated by Cosminexus HTTP Server is added to the Set-Cookie header, flow restriction is performed by using the cookie. Requests with the cookie are processed with a higher priority than requests without cookies.

If you specify the following, and if the number of requested server threads is 10, the server will deny a new session request without cookie, however, the server will process a maintenance session with cookie. If the number of requested server threads is 13, the server will deny the request regardless of the cookie. This example is performed in the HWS creation mode.

- UNIX version

```
MaxClients 15
QOSRejectionServers 2
```

```
QOSCookieServers 5
```

- Windows version

```
ThreadsPerChild 15
QOSRejectionServers 2
QOSCookieServers 5
```

(c) Redirect

When the flow-restricting functionality denies a request process the server returns a response message with the status code 503. However, you can redirect the request to some other Web server by specifying the following. If a request for /index.html is denied from the flow-restricting functionality, index.html of the www1.hitachi.co.jp Web server is set in the response header, and the request is returned with the status code 302:

```
QOSRedirect /index.html http://www1.hitachi.co.jp/index.html
```

(d) Customizing the response message

If you specify the following, the contents of htdocs/busy.html are returned in the response message with the status code 503, as the response for the denied request.

```
QOSResponse file "text/html; charset=ISO-8859-1" htdocs/
busy.html
```

(3) Response message

(a) Cookie sent by the server to the client

HWS creation mode

The cookie generated by Cosminexus HTTP Server is returned to the client by using the Set-Cookie header. Because multiple Set-Cookie headers can be specified in a response, the cookies generated here do not affect the other cookies. The following shows the Set-Cookie header returned by Cosminexus HTTP Server:

```
Set-Cookie: NAME=VALUE; expires=DATE; path=/; domain=DOMAIN_NAME; secure
```

NAME=VALUE

The name specified in the QOSCookieName directive is set as NAME. The value for the request control is specified for VALUE.

expires=DATE

Specifies the time when cookie becomes invalid. The value acquired from "received request time + setting value of QOSCookieExpires directive" is set with RFC822 format.

path=/

A URL to which the cookie is applied. In this module, settings are done so that the cookie will be valid for all the URLs in the domain for which the cookie is valid.

domain=DOMAIN_NAME

Domain to which the cookie is applied. The value is specified in QOSCookieDomain directive.

secure

Specify whether to send **cookie** from the client to the server when using only SSL. The value is specified in QOSCookieSecure directive.

User creation mode

Cosminexus HTTP Server does not generate cookies. Cookies generated by programs other than Cosminexus HTTP Server are returned to the client by using the Set-Cookie header.

(b) Headers when access is denied by the flow-restricting functionality

When access is denied by the flow-restricting functionality, the Expires header, which stops the response message from being cached, is added to the response header. This is because if the response is cached, the cached message is displayed, and requests might not be sent to the server, even if the server can accept request processing. The server disconnects after sending the denial message.

The other response headers are set as shown below. The character set specified in the AddDefaultCharset directive is added to the Content-Type header.

(i) Standard message for the status code 503

Content-Type: text/html

(ii) A customized message from QOSResponse

Content-Type: The QOSResponse directive specification value

(iii) A Message for the status code 302 from QOSRedirect

Content-Type: text/html

Location: The QOSResponse directive specification value

(4) Notes

- If a client denies reception of a cookie, the cookie is not sent back to the server, so the functionality of (2)(b) Session management using cookies is disabled.
- For a KeepAlive connection the determination process is executed only when

processing the first request after connection, and the determination process is not executed from the second request onwards in the same connection. If the request of flow-restriction settings differs with the request that is connected initially, the server does not determine the request from second request onwards.

- The message sent when request is denied by flow-restricting functionality will not change even if you specify the ErrorDocument directive. Also, the Redirect directive and RedirectMatch directive are executed after mod_hws_qos module control determines that processing can continue.
- If the server simultaneously receives requests that exceed number of settings of QOSRejectionServers directive, the server may not deny the requests correctly even if you use the flow-restriction functionality. If a HTML file specified in QOSResponse directive contains a link such as image data link, the server accesses again to fetch the image data. This access also becomes the target of flow-restriction process, and you may not obtain the image data. Be careful to set the proper link when you create HTML file.
- Setting of HWSErrorDocumentMETACharset directive is valid in Windows version for the character set of error document.
- You must set another name in QOSCookieName directive if you set the session management in which cookie is used for each URL, or for VirtualHost.
- Set the QOSCookieServers directive and QOSRejectionServers directive after setting the number of server processes (set in the ThreadsPerChild directive or MaxClients directive). If you set the QOSCookieServers directive and QOSRejectionServers directive before setting the number of server processes, the server may not start.

4.10 Header customization functionality

With HTTP communication, various HTTP headers are used between the Web browser and the Web server. The Web browser and the Web server sometimes determine the subsequent operations by using the received HTTP headers. The HTTP headers added by the Web browser at the time of sending an HTTP request is called the *request header*, and the HTTP header added by the Web server in the response is called the *response header*. The functionality that is used for the Web server or the Web browser to perform a specific operation by adding, changing, or deleting the request header, which is received or sent by the Web server, is called the *header customization functionality*.

You can use header customization functionality by embedding the mod_headers module in Cosminexus HTTP Server.

(1) Embedding the mod_headers module

You must embed the mod_headers module to use the header customization functionality. Specify the following directive in configuration file (`httpsd.conf`) to embed the mod_headers module:

- UNIX version

```
LoadModule headers_module libexec/mod_headers.so
```

- Windows version

```
LoadModule headers_module modules/mod_headers.so
```

(2) How to set directives

The header customization functionality is specified in the Header directive or the RequestHeader directive. The following shows examples of setting the directive to use the header customization functionality.

(a) When setting the response header

The response header can be set by using the set indicator of the Header directive. If the response header with the same name is already set in another module, the header value is overwritten.

The following is an example in which Expires: Sat, 1 Jan 2000 00:00:00 GMT is set in the response header. However, to set the expiry date dynamically, use the expiry date settings functionality:

```
Header set Expires "Sat, 1 Jan 2000 00:00:00 GMT"
```

(b) When adding the response header

The response header can be added by using the add indicator of the Header directive. Even when the response header with the same name is already set in another module, the header is added as an additional header. The `add` indicator is used to set response headers with the same name on multiple lines.

The following is an example to add Set-Cookie: HOSTNAME=HOST1; path=/; domain=www.example.com; secure in response header:

```
Header add Set-Cookie "HOSTNAME=HOST1; path=/; domain=www.example.com;
secure"
```

(3) Note

- The Date, Server, Content-Type, Content-Length and Last-Modified headers of response headers might not be customized. In addition, if another module is embedded by using the LoadModule directive, the headers other than the above headers also might not be customized.

4.11 Functionality to set expiry date

When you set the expiry date in the Web server contents, the client and the proxy server supporting the cache functionality do not access the Web server during that period and access their own cache, and hence this functionality is effective.

You can use expiry date setting functionality by embedding the mod_expires module in Cosminexus HTTP Server. You can use the expiry date setting functionality to execute the following operations:

- If you use the expiry date settings functionality, the Expires header and the Cache-Control header are added to response.
- With the Expires header the expiry date is set in the Greenwich standard time (GMT), and with the Cache-Control header the time up to the expiry date is set in seconds in the max-age directive.

The client and the proxy server handle the set Expires header and the Cache-Control header.

(1) Embedding the mod_expires module

You need to embed the mod_expires module to use the expiry date setting functionality. Set the following directive in configuration file (`httpsd.conf`) to embed the mod_expires module:

- UNIX Version

```
LoadModule expires_module libexec/mod_expires.so
```

- Windows Version

```
LoadModule expires_module modules/mod_expires.so
```

(2) How to set directives

The following are examples of setting the directive to use the expiry date setting functionality:

(a) Setting the default expiry date

Set the default expiry date using the ExpiresDefault directive for all contents on the Web server. Set the expiry date on the basis of file update time or the time when client accessed.

If you set the following, the Expires header and the Cache-Control header are added to the response by considering the validity period as the time after `60` seconds from the client access time:

```
ExpiresActive On
ExpiresDefault A60
```

"A" specified in the ExpiresDefault denotes that the client access time is the standard time.

(b) Setting the expiry date for each MIME

Set the expiry date for each MIME type with the ExpiresByType directive. The default expiry date set by the ExpiresDefault directive is overwritten for each MIME type by these settings. Set the expiry date on the basis of file update time or the time when client accessed.

If you specify the following, the Expires header and the Cache-Control header are added to the response by considering the validity period as the time after one hour from the file update time, only when the MIME type is text/html:

```
ExpiresActive On
ExpiresByType text/html M3600
```

"M" specified in the ExpiresByType denotes that the file update time is the standard time.

(3) Notes

- When setting the file update time as the standard time, the update time is not available for the requests (requests that display status information) that do not access the files on the disk, so the Expires header and the Cache-Control header are not added.
- If you embed a module that is not provided as a standard module by Cosminexus HTTP Server into the LoadModule directive, the Expires header and the Cache-Control header may get operated.
- If you use the header customization functionality simultaneously, do not operate the Expires header and the Cache-Control header with the header customization functionality.

4.12 Static contents cache functionality

You can cache the static contents file stored on the disk in the memory, return the file from the cache to the browser, and can reduce the response time of static contents.

You can use this functionality only in Windows version.

(1) Embedding the module

Embed the mod_hws_cache module to use the static contents cache functionality. Specify the configuration file (`httpsd.conf`) as follows to embed the mod_hws_cache module:

```
LoadModule hws_cache_module modules/mod_hws_cache.so
```

(2) How to set directives

Specify the handler name hws_cache to use the cache functionality in the AddHandler directive or the SetHandler directive.

(a) Settings to cache the file with a specific extension

Set the AddHandler directive to cache the file with a specific extension. The setting method is as follows:

```
<Directory "<Cosminexus-installation-directory>/httpsd/htdocs">
  AddHandler hws_cache .html
</Directory>
```

(b) Settings to cache the file under a specific URL

Set the SetHandler directive to cache the file under a specific URL. The setting method is as follows:

```
<Directory "<Cosminexus-installation-directory>/httpsd/icons">
<Files "?*">
  SetHandler hws_cache
</Files>
</Directory>
```

(3) Triggers of file cache, cache update, and cache deletion

(a) Trigger to cache a file

The trigger to cache a file is shown below. However, if the file size is larger than the size set in the HWSContentCacheMaxFileSize directive, the file is not cached.

- When the file for caching is requested.

(b) Trigger to update the cache

The trigger in which cache is updated is shown below. However, if the file size is bigger than the size set in the HWSContentCacheMaxFileSize directive, the file is not cached.

- When the update time of cached file is changed.

(c) Trigger to delete the cache

The trigger in which cache is deleted is shown below:

- When the total size of cached file exceeds the size specified in the HWSContentCacheSize directive (in this case, the cache is deleted from the cache file that is not requested for a long time).
- When the Web server is restarted.

(4) Memory usage

You can calculate the maximum memory size used in the static contents cache functionality with the following formula:

Formula

```
(HWSContentCacheMaxFileSize × ThreadsPerChild) +
HWSContentCacheSize
```

(5) Note

Do not cache the files placed in the URL in which the reverse proxy, CGI, and other dynamic content processes are performed. If you specify files as files to be cached by mistake, dynamic content might not be processed correctly.

The following is a usage example:

```
ScriptAlias /cgi-bin/ "<Cosminexus-installation-directory>/httpsd/cgi-bin/"
                                                               *1
LoadModule hws_cache_module modules/mod_hws_cache.so
<Directory "<Cosminexus-installation-directory>/httpsd/icons/">
                                                  *2
<Files "?*">
  SetHandler hws_cache
</Files>
</Directory>
```

# Do not duplicate the path that is set in underlined *1 and *2.

When "<*Cosminexus-installation-directory*>/httpsd/cgi-bin/" is set as *2, the CGI program is responded as static file, and you may get unexpected results such that contents of CGI programs are leaked.

4.13 Generating multiple Web server environment (hwsserveredit command)

If multiple Web servers are running on a single server machine and not in the virtual host, you need to set the environment such as preparing the `httpsd.conf` on each Web server. The hwsserveredit command supplements this environment setting.

(1) Format

```
hwsserveredit {-add|-delete|-check} server-name
```

(2) Parameters

- -add

  Specify this parameter when you create a new server environment. If the command receives a request, create a directory with the same name as the specified server name, and then create the `httpsd.conf` and logs directory. Additionally, for Windows version, use the server name and register Cosminexus HTTP Server as a service.

- -delete

  Specify this parameter to delete a server environment. If the command receives a request, this parameter deletes the directory with the name same as the server name under the server directory. For Windows version, this parameter deletes the service of server name.

- -check

  Specify this parameter to confirm that the server environment is built. If the command receives a request, and if there is a resource that is created when the `-add` parameter is requested, judge that the server startup environment is built.

- *server-name* ~ ((1-(220-path length of Cosminexus HTTP Server installation directory), or 128 bytes or below))

  Specify a unique character string for each server. However, you cannot specify "Cosminexus HTTP Server". Also, you cannot specify the character string "Cosminexus HTTP Server" (such as "Cosminexus HTTP Server") formed by removing the spaces in the character string.

(3) How to use

(a) Creating the resource

Determine the server name of each server, and execute the hwsserveredit command.

Specify the following, if the server name is *HWS1*:

```
hwsserveredit -add HWS1
```

The hwsserveredit command creates directories and files under the server directory that is created when installing the server. In Windows version, the utility registers the service using server name at the same time when creating directories and files.

The configuration of directories and files are as follows:

```
+httpsd
   +servers
       +HWS1
          +conf
          | +httpsd.conf
          +logs
```

(b) Editing the httpsd.conf

Change the `httpsd.conf` directive value in the created resources.

When you generate multiple environments, change the values of following directives so that the environment does not clash with other environments:

- ServerName directive
- Port directive or Listen directive

For the UNIX version, change the setting value of following directives according to the environment:

- User directive
- Group directive

Other directives are set in such a way that they can run even if you do not change their settings depending upon the hwsserveredit command. Depending upon the operations, change the settings if required.

(c) Starting up the server

Start up the server by following method:

- For UNIX version
  `/opt/hitachi/httpsd/sbin/httpsd -f servers/HWS1/conf/httpsd.conf`
- For Windows version
  `"<`*`Cosminexus-installation-directory`*`>\httpsd.exe" -n HWS1 -k`

```
start
```

Moreover, start up the HWSI service from the control panel.

(d) Setting up the multiple server environments

To set the multiple server environments repeat operations from (a) to (c).

(4) Notes

- Run the command as a user with administrator permissions. Do not move the command location.
- Specify server name with the ASCII codes. Note that you cannot specify the following characters:

  '￥', '/', ':', ',', ';', '*', '?', '"', '<', '>', '|', '$', '%', '^', ''', '!', '(', ')', '=', '+', '{', '}', '@', '[', ']', '~', and control codes
- In a server name, you cannot specify a name that consists of only a period. Additionally, if there is an immediate space before or after the name, remove the space.
- When registering the service, register the startup type manually.
- Do not change the display name of registered service.

4.14 Image map

You can define multiple links in an image (image file). If you click on the specified point, the coordinate position and the image file name are sent from the Web browser to the Web server. The Web server searches for a URL that corresponds to the image map file and coordinate position, and responds to the Web browser. This is called *image map*.

To use the image map, you need to define mapping file extension in the imap-file handler.

```
AddHandler imap-file .map
```

(1) Syntax of image map file

You can specify the image map data in following three formats:

```
shape name specification value coordinate
shape name specification value "descriptive text" coordinate
shape name specification value coordinate "descriptive text"
```

"descriptive text" denotes the explanation sentence when map file menu is displayed and coordinate denotes the coordinates of images.

Table 4-15 describes the shape name and table 4-16 describes the specification value:

Table 4-15: Specification format of shape name and coordinates

Shape name	Meaning	Specification of coordinates	Explanation of coordinates
base	Specifies base of a relative URL in a map file.	None	--
default	Specifies a link, when figure name is not related to poly, circle, and rect, and point is also not specified.		
poly	Specifies a polygon having 3 to 100 sides.	x1,y1 x2,y2 ... xn,yn	Every coordinate location of polygon (coordinate for 3 to 100 sides)

Shape name	Meaning	Specification of coordinates	Explanation of coordinates
circle	Specifies a circle. The circle specifies a center point and one point on circumference.	x1,y1 x2,y2	Coordinates of center point and one point on circumference
rect	Specifies a rectangle. The rect specifies 2 points of opposite corners.	x1,y1 x2,y2	Coordinates of 2 points of opposite corners
point	Specifies a point. The point nearest to the cursor is valid.	x1,y1	Point

Legend:

--: Not applicable.

#

Even if (0,0) is included in the coordinates specification, if you point the coordinates (0,0) of image map image by mouse pointer, the map file menu is displayed.

Table 4-16: Specification values

Specification values	Meaning
URL	Specifies the link destination. The base and ImapBase directives are enabled in the case of a relative directory.
map	Displays a map file menu.
menu	
referer	Responds with the status code 302 Found.
nocontent	Responds with the status code 204 No Content. Valid for other than base.
error	Responds with the status code 500 Server Error. Valid for other than base.

(2) Example of image map definition

The operations to use the image map are as follows:

1. Set the following directives in the `httpsd.conf` file. Execute the image map when `.map` extension name is specified in URL.
```
AddHandler imap-file .map
```

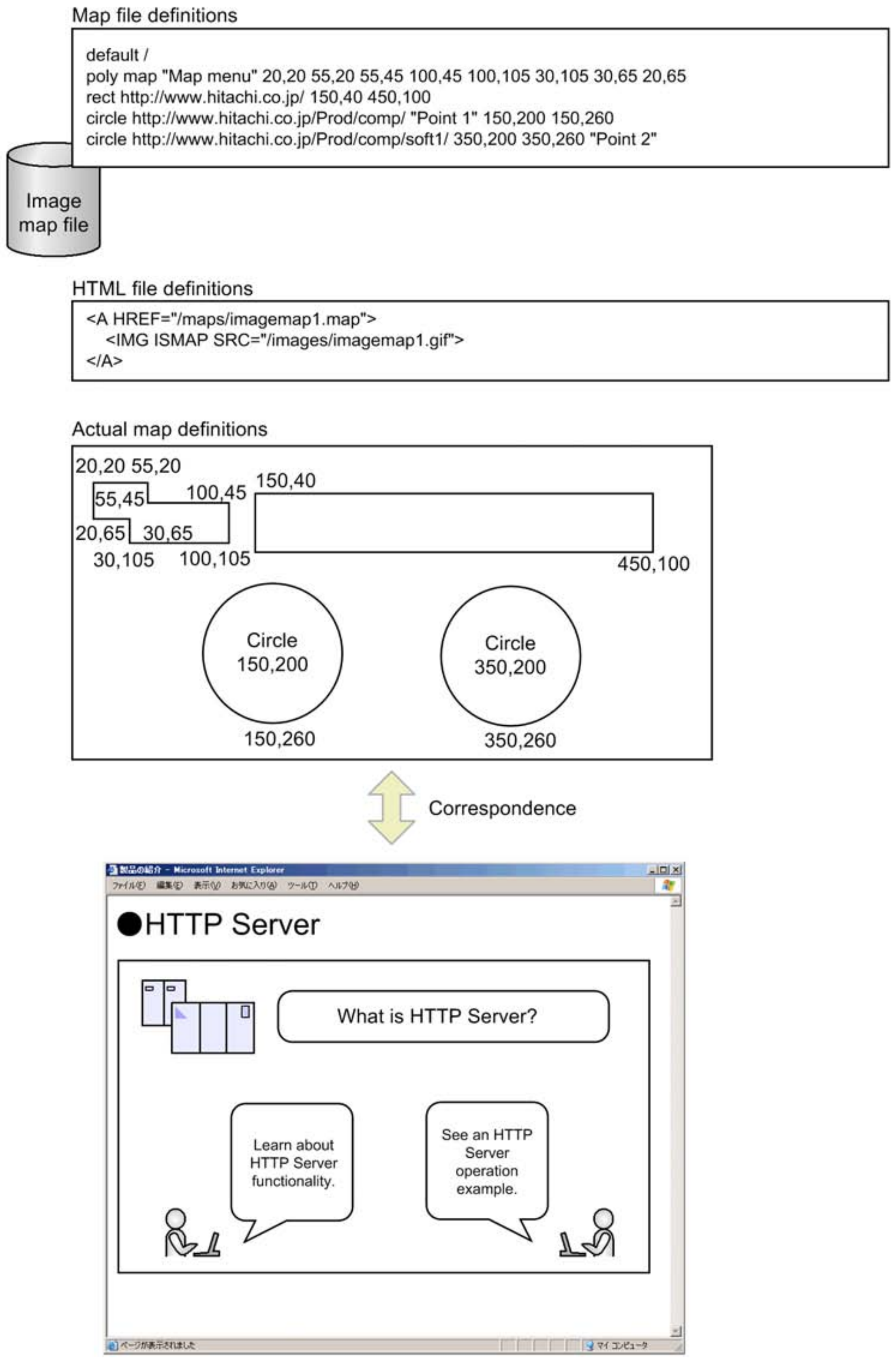
(Define the imap-file handler in the file extension `.map`)

2. Define a link destination in the file with the extension defined above.
3. Describe the following HTML syntax in HTML document.

```
<A HREF="/directory name/map file name"><IMG SRC="image data
name" ISMAP></A>
```

The following figure shows an example of image map file definition and actual display:

Figure 4-13: Example of image map filedefinition



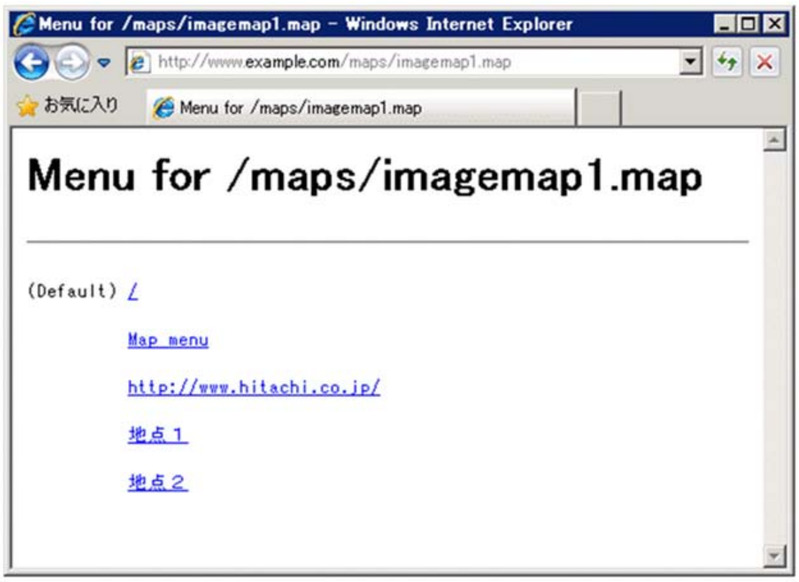
Map file definitions

```
default /
poly map "Map menu" 20,20 55,20 55,45 100,45 100,105 30,105 30,65 20,65
rect http://www.hitachi.co.jp/ 150,40 450,100
circle http://www.hitachi.co.jp/Prod/comp/ "Point 1" 150,200 150,260
circle http://www.hitachi.co.jp/Prod/comp/soft1/ 350,200 350,260 "Point 2"
```

HTML file definitions

```
<A HREF="/maps/imagemap1.map">
   <IMG ISMAP SRC="/images/imagemap1.gif">
</A>
```

In this example, if you click on the part specified with poly, the following map file menu is displayed:



Menu for /maps/imagemap1.map - Windows Internet Explorer
http://www.example.com/maps/imagemap1.map
お気に入り
Menu for /maps/imagemap1.map

Menu for /maps/imagemap1.map

(Default) /

Map menu

http://www.hitachi.co.jp/

地点１

地点２

(3) Note

If the character set used in the map file menu differs from the default character set (ISO-8859-1), characters will become garbled in the map file menu display. In this case, specify the character set used in the map file menu in the HWSImapMenuCharset directive.

4.15 IPv6 connections

Not only existing IPv4 connections but also IPv6 connections are allowed. Connections can be performed in an IPv4 environment or an IPv4/IPv6 dual-stack environment.

4.15.1 Support range

(1) Directives that support IPv6

Specifying IPv6 addresses in the Listen directive and the VirtualHost directive allows connections via IPv6, and the specification of virtual hosts supporting IPv6 addresses.

The following directives support IPv6:

<VirtualHost>, AddIcon, AddIconByEncoding, AddIconByType, Allow from, CustomLog, DefaultIcon, Deny from, ErrorDocument, ExtendedStatus, HostnameLookups, HWSErrorLogClientAddr X-Forwarded-For, HWSSetEnvIfIPv6, ImapBase, ImapDefault, Listen, LogFormat, NameVirtualHost, ProxyPass, ProxyPassReverse, QOSCookieDomain, QOSRedirect, Redirect, RedirectMatch, ServerAlias, ServerName, ServerSignature, SetEnvIf, SetEnvIfNoCase, TransferLog, UseCanonicalName

(2) Note on specifying IPv6 addresses in directives

When an IPv6 address is specified in a directive, enclose the IPv6 address in square brackets ([]), as in the case of [*IPv6-address*]. When an IPv6 address and a port number are specified in a directive, enclose the IPv6 address in square brackets ([]) and specify the port number after a colon (:), as in the case of [*IPv6-address*]:*port-number*.

However, do not enclose the IPv6 address in square brackets ([]) when IPv6 addresses are specified in the following directives:

- Allow from directive
- Deny from directive
- HWSSetEnvIfIPv6 directive

Specify a global unicast address when using an IPv6 address.

(3) Limitations

The following functions do not support IPv6:

Some SSL communication

When the port number is specified in the SSLCacheServerPort directive, the IPv4 connection is performed between the Web server and the gcache server. IPv6 connections cannot be performed.

Note that an SSL connection with an IPv6 socket can be performed.

Directory service (LDAP server)

An IPv6 address or the host name corresponding to the IPv6 address cannot be specified in the LDAPServerName directive.

Address limitation

An IPv6 address cannot be specified in the BindAddress directive.

Client check

The client is not checked when the IPv6 socket is used even if the IdentityCheck directive is set to On.

Environment variable settings

An IPv6 address cannot be specified for regular expressions in the SetEnvIf directive and the SetEnvIfNoCase directive. Use the HWSSetEnvIfIPv6 directive when using an IPv6 address.

crldownload command

An IPv6 address and the host name corresponding to the IPv6 address cannot be specified in `-h` *host-name*.

4.15.2 Preparing for an IPv6 connection (Editing the httpsd.conf file)

Settings for using IPv6 addresses

When the Port directive or the Listen directive which has only a port number is specified in the httpsd.conf file, only requests via IPv4 addresses are accepted. When you want to use IPv6 addresses, you must set the Listen directive in which an IPv6 address is specified.

For example, the following settings enable requests via IPv4 addresses or IPv6 addresses to be accepted:

```
Listen 80
Listen [::]:80
```

4.16 Integration with an application server

For integration with an application server, see the *uCosminexus Application Server Web Container Functionality Guide*.

Chapter

5. Authentication and Encryption by Using SSL

This chapter describes the authentication and encryption by SSL.

5.1 Authenticating and encrypting with SSL

If you use the Secure Sockets Layer (SSL) protocol, Cosminexus HTTP Server can secure the sending and receiving of information. The Cosminexus HTTP Server supports SSL version 3, Transport Layer Security (TLS) version 1.0, 1.1 and 1.2, and can authenticate SSL servers and clients. The SSL features are as follows:

- Checking and authenticating the communication partner identity.
- Encrypting the data to be transferred between the server and the client.
- Detecting the tampered data during the data transfer.

You can use these features if you install a private key created by an SSL-related command and a certificate issued by the Certification Authority (CA) on the Web server.

5.1.1 Preparing for SSL communication

You need to install the private key and the certificate issued by the Certification Authority (CA) on the Web server to use the authentication and the data encryption by SSL.

Perform the following:

1. Creating a private key

   Use the `keygen` command to create a private key for the Web server.

2. Creating a Certificate Signing Request (CSR)

   Use the `certutil reqgen` command to create a Certificate Signing Request (CSR).

3. Sending the CSR to CA

   Send the CSR created in the above step 2 to the CA.

4. Acquiring a certificate

   Acquire a PEM formatted certificate from the CA.

5. Editing the `httpsd.conf` file (defining directives)

   Specify the SSLEnable directive to enable the SSL. Specify the PEM formatted certificate acquired from the CA in the SSLCertificateFile directive and private key of the Web server in the SSLCertificateKeyFile directive.

   Example: This example enables SSL and defines the PEM formatted certificate and Web server private key.

   - For UNIX Version

```
SSLEnable
SSLCertificateFile /opt/hitachi/httpsd/conf/ssl/server/httpsd.pem
SSLCertificateKeyFile /opt/hitachi/httpsd/conf/ssl/server/httpsdkey.pem
```

- For Windows Version

```
SSLEnable
SSLCertificateFile "<Cosminexus-installation-directory>/httpsd/conf/ssl/
server/httpsd.pem"
SSLCertificateKeyFile "<Cosminexus-installation-directory>/httpsd/conf/ssl/
server/httpsdkey.pem"
```

When communicating with SSL, use `https://` to request from the Web browser. When you omit the port number, the Web server uses the port 443 with standard SSL. However, it is common to specify port 443 in the Port or Listen directive.

6. Restarting the Web server

   Restart the Web server to enable the definitions of the `httpsd.conf` file. However, if you change the settings of the SSLCertificateKeyFile directive, stop and then restart the Web server.

To disable SSL, disable the specification given in the above-mentioned step 5, specify the SSLDisable directive and then restart the server.

5.1.2 Procedure of SSL communication

The procedure of SSL communication is described below. The procedure mentioned in steps 2 to 6 below is called SSL *handshake*:

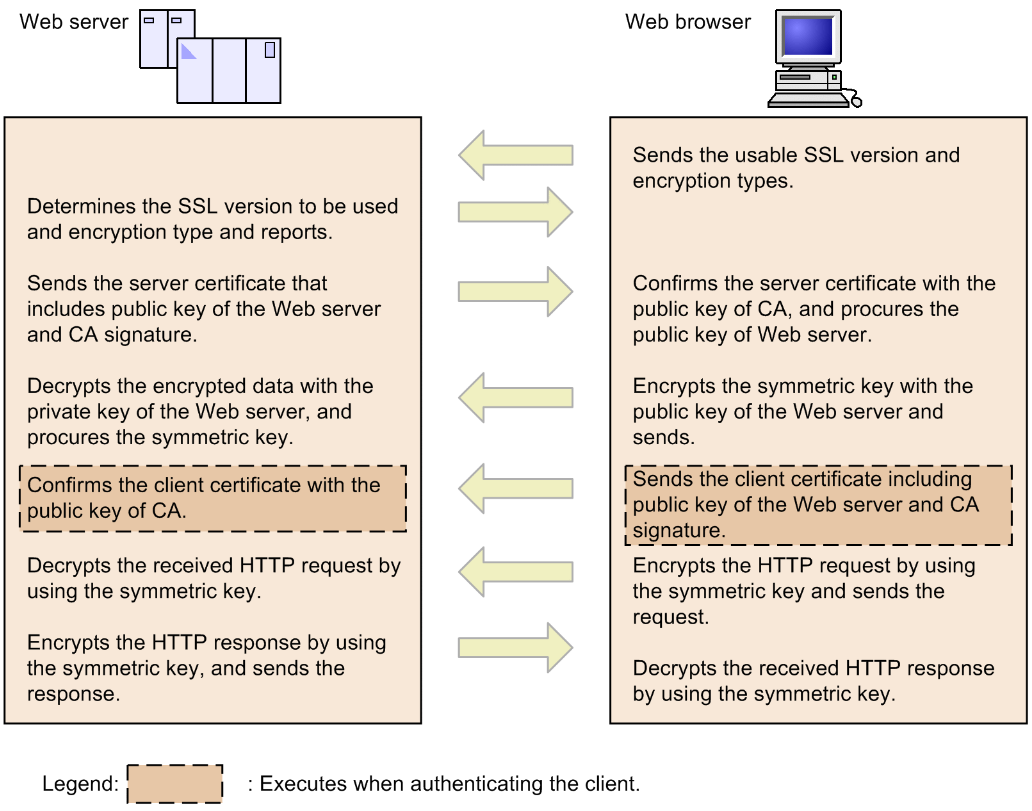
1. Execute `https://` requests from the Web browser.
2. The Web browser sends SSL versions that can be used and the data that displays encryption types to the Web server.
3. The Web server determines and reports the SSL version to be used and the encryption type to the Web browser. Additionally, the Web server sends the public key and the certificate with CA signature to the Web browser.
4. The Web browser uses an available CA public key, confirms that the sent certificate is not tempered, and then acquires Web server public key.
5. The Web browser creates a symmetric key that is shared with the Web server through communication, encrypts the symmetric key with Web server public key, and then sends. To present the available certificate for client authentication on the Web server, the Web browser sends the certificate to the Web server.

The data encrypted by Web server public key cannot be decrypted, if there is no corresponding private key. In other words, only the Web server to which the data is sent can decode the data items.

6. The Web server decrypts the received symmetric key with Web server private key, and procures the symmetric key. When the Web server receives the certificate from the Web browser, the Web server confirms the certificate.
7. The HTTP request or the response is encrypted for a two-way transmission, with the symmetric key shared between the Web browser and the Web server.

The following figure shows the SSL communication request process.

Figure 5-1: SSL communication request process


Web server
Web browser
Sends the usable SSL version and encryption types.
Determines the SSL version to be used and encryption type and reports.
Sends the server certificate that includes public key of the Web server and CA signature.
Confirms the server certificate with the public key of CA, and procures the public key of Web server.
Encrypts the symmetric key with the public key of the Web server and sends.
Decrypts the encrypted data with the private key of the Web server, and procures the symmetric key.
Sends the client certificate including public key of the Web server and CA signature.
Confirms the client certificate with the public key of CA.
Encrypts the HTTP request by using the symmetric key and sends the request.
Decrypts the received HTTP request by using the symmetric key.
Encrypts the HTTP response by using the symmetric key, and sends the response.
Decrypts the received HTTP response by using the symmetric key.
Legend:
: Executes when authenticating the client.

The strongest available encryption valid for both the client and the Web server is chosen for handshaking. You use the SSLRequiredCiphers directive to specify the encryption type of the Web server. If you always enable all the encryption types with this specification, you can use the strongest encryption available on the client to send the data.

5.1.3 SSL session management

Many Web browsers are currently implementing functionality that uses SSL sessions to simplify the handshake. Use the browser functionality and the SSL session management functionality to improve the efficiency of the SSL communication efficiency.

SSL session management method differs in UNIX and Windows versions. In UNIX version, a management server (*gcache server*) is used to manage the SSL session. The gcache server receives the information about the SSL session ID, validity period, and the session through the specified port and file, and manages the session. You can use the gcache server to share the data such as SSL session IDs between request processes of Cosminexus HTTP Server. In Windows version, the SSL session is managed based on the structure of the Web server without using the gcache server.

(1) For UNIX Version

(a) Starting and stopping the gcache server

The gcache server starts if you enable SSL and specify directives that are required to start the gcache server, and start Cosminexus HTTP Server. SSL is enabled when either the SSLEnable directive is specified, or there is a host on which SSLDisable directive is not specified (including virtual host).

You need to specify the following directives to start the gcache server:

- SSLCacheServerPath
- SSLCacheServerPort

Note that if you stop the Web server, the gcache server also stops at the same time. If you restart the Web server, the gcache server stops once, and then restarts.

(b) Session management area

If you establish the SSL session, the information of that session is cached in the gcache server and Web server process. You can specify cache area size of the gcache server in the SSLSessionCacheSize directive, and the cache in the Web server process in the SSLSessionCacheSizePerChild directive.

When you set the SSLSessionCacheSize directive to `0`, the SSL session is not managed.

The validity period of the SSL session is the time period that is lesser of two values; the value specified in the SSLSessionCacheTimeout directive, or the time taken by the cache size to reach the specified value of the SSLSessionCacheSize directive.

When the cache size reaches the value specified for the SSLSessionCacheSize directive, the oldest session information is deleted until memory sufficient to store new session information is secured.

You can reuse the cached session information to simplify the SSL handshake at the time of establishing the next session.

(c) Notes

1. The data such as the session ID is cleared if you stop the Web server or the gcache server.
2. When gcache server is stopped abnormally, the SSL session cannot be sustained. However, the Web service does not stop.
3. Web servers cannot share the gcache server for security reasons. Therefore, the SSL session cannot be shared between Web servers.
4. The gcache server does not start, if you set the SSLSessionCacheSize directive to `0`.
5. The session cache area is not divided for each virtual host.

(2) For Windows Version

(a) Session management area

If you establish the SSL session, the session information is cached in Web server process. You can specify the cache area size in the SSLSessionCacheSize directive. When you set the SSLSessionCacheSize directive to `0`, the SSL session is not managed.

The validity period of the SSL session is the time period that is lesser of two values; the value specified in the SSLSessionCacheTimeout directive or the time taken by the cache size to reach the specified value of the SSLSessionCacheSize directive.

You can reuse the cached session information to simplify the SSL handshake at the time of establishing the next session.

(b) Notes

- The data such as the session ID is cleared if you stop the Web server.
- The session cache area is not divided for each virtual host.

5.1.4 Preparing for SSL client authentication

For authenticating the SSL client, execute the following operations in addition to steps 1 to 5 of *5.1.1 Preparing for SSL communication*, and then restart the Web server:

1. Installing the client certificate for the Web browser

   Install the client certificate for the Web browser as per the CA instructions used to issue certificate.

2. Acquiring the CA certificate

   Acquire the certificate (PEM formatted) of CA that has issued the client certificate.

3. Editing the `httpsd.conf` file (definition of directives)

   To enable the SSL client authentication, specify `2` in the SSLVerifyClient and `2` or more in the SSLVerifyDepth directive. Specify the PEM formatted certificate acquired from the CA in the SSLCACertificateFile or SSLCACertificatePath directive.

   #

   You cannot specify the SSLCACertificatePath directive in Windows version.

5.1.5 Verifying the validity of certificates

When the Web server authenticates the SSL client, the Web server can use the Certificate Revocation List (CRL) to verify the client certificate and also to verify the validity of the client certificate at that time. Acquire the CRL from the CA that issues the client certificate to be verified.

(1) CRL file format

The CRL uses the DER formatted file or the PEM formatted file. The CRL in the DER format is a file in the binary format and the CRL in the PEM format is a file in the Base64 encoded format with"-----*BEGIN X509 CRL*-----", "-----*END X509 CRL*-----" tags attached before and after the data.

Example: CRL in the PEM format

```
<Cosminexus-installation-directory>\httpsd\conf\ssl\crl\PEM>type
crl.pem
-----BEGIN X509 CRL-----
MIIBGDCBwwIBATANBgkqhkiG9w0BAQQFADB0MQswCQYDVQQGEwJKUDERMA8GA1UECBM
IS2FuYWdhd2ExFTATBgNVBAcTDFlva29oYW1hLXNoaTERMA8GA1UEChMITE9D
QUwtQ0ExDDAKBgNVBAsTA2NhMTEaMBgGA1UEAxMRY2ExLmhpdGFjaGkuY28uanAX
DTAxMDgyOTA0NDIzMFoXDTAxMDgzMDA1NTIzMFowGzAZAghx2Sa8AAAAARcNMDEw
ODI4MDQ1MTI5WjANBgkqhkiG9w0BAQQFAANBAJorY7DUJ91uthNlAA+PT6zw6rVo
uZLFeYZPNVXgF217YOCtJtKDT+16bR5kgk0p/1xIbgReshjMNTmXPqARNjE=
-----END X509 CRL-----
```

(2) CRL application method of Cosminexus HTTP Server

When you use the CRL to verify the validity of the client certificate, execute the following steps in addition to steps mentioned in *5.1.4 Preparing for SSL client authentication*, and restart the Web server:

1. Acquiring CRL

   Acquire the CRL file from the CRL distribution points of each CA and store in the appropriate directory. For managing the CRL in the LDAP server, you can use the crldownload command.

2. Editing `httpsd.conf` (directive definition)

   To enable the CRL, specify the directory where CRL file is stored in the SSLCRLDERPath or SSLCRLPEMPath directive.

3. Start or restart the Web server.
4. When updating the existing CRL, delete the old CRL that is stored in the directory and add a new CRL, or overwrite the old CRL by the new one, and then restart the Web server.
5. When adding a new CRL and also when deleting a CRL, restart the Web server.

(3) Verifying the client certificate that uses the CRL

Confirm the following items to verify the client certificate that uses the CRL:

- Whether CRL is valid.
- Whether date is before the next issue date.
- Whether the serial number of the client certificate is mentioned.

(a) Verifying the CRL client certificate and determining the client certificate as valid

The following are the conditions when the client certificate is determined as valid by verifying the CRL client certificate:

- When the CA that issues the certificate does not read the issued CRL.
- When the current date is before the next issue date, and the serial number of the corresponding connected client is not mentioned in the CRL.
- When the current date is after the CRL issue date, the next issue date is not specified, and the serial number of the connected client certificate corresponding to the CRL is not mentioned.
- When the current date is after the next issue date of the CRL, the serial number of the connected client certificate corresponding to the CRL is not mentioned, and when the SSLCRLAuthoritative directive is set to `off`.

(b) Verifying the CRL client certificate and determining the client certificate as invalid

The following are the conditions when the client certificate is determined as invalid during the CRL client certificate verification:

- When the CRL is invalid.
- When the serial number of the client certificate of corresponding connection is mentioned in the CRL.
- When the current date is after the CRL next issue date, the serial number of the

client certificate of corresponding connection is not mentioned in the CRL, and when the SSLCRLAuthoritative directive is set to `On`.

5.2 Acquiring a certificate

The following figure shows the procedure to acquire a certificate.

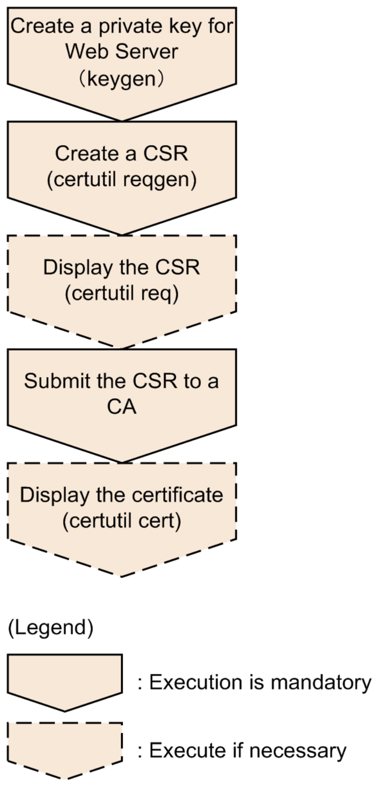

Create a private key for Web Server (keygen)
Create a CSR (certutil reqgen)
Display the CSR (certutil req)
Submit the CSR to a CA
Display the certificate (certutil cert)
(Legend)
: Execution is mandatory
: Execute if necessary

Figure 5-2: Acquiring the certificate

Acquire a certificate file signed by the CA according to the procedure shown in the above figure, and then save the part of the certificate file that begins with `-----BEGIN CERTIFICATE-----` and ends with `-----BEGIN CERTIFICATE-----` into a separate file (`httpsd.pem` in the standard `httpsd.conf`). After that, define the file in the SSLCertificateFile directive to use SSL.

- keygen

  A command that creates the Web server private key required to acquire the certificate when using SSL

- certutil

  A command that creates the Certificate Signing Request (CSR) required to acquire the certificate when using SSL

5.2.1 Creating a private key for the Web server (keygen command)

This subsection describes how to use the `keygen` command to create a private key for the Web server. The created Web server private key file is specified in the SSLCertificateKeyFile directive.

(1) Format

```
keygen -rand file-name[:file-name...] [-des|-des3]-out key-file [-bits
{512|1024|2048|4096}]
```

(2) Parameters

- `-rand` *file-name* `[:`*file-name*`...]`

  Specify any file to be used for random number generation. You must specify an appropriate file whose size is large enough for the random number generation (for example, `C:\WINNT\NOTEPAD.EXE`). In Windows, you can specify only one file name. Multiple file names cannot be specified.

- `[-des|-des3]`

  Specify the encryption type when encrypting the private key. If you specify this parameter, you will be requested to enter a password when creating the private key. The password must be no more than 64 characters long.

  When creating the Certificate Signing Request (CSR) (see *5.2.2 Creating a Certificate Signing Request (CSR) (certutil reqgen command)*) and starting the Web server, you will also be requested to enter the password. Note that you can skip the password entry for Web server startup (see *5.4 Using the server private key with a password*). If `-des` is specified, the Data Encryption Standard (DES) is selected as the encryption type. If `-des3` is specified, Triple DES is selected. This parameter does not affect the encryption type used in the communication between the Web server and the Web browser.

- `-out` *key-file*

  Specify the file to which the Web server private key is output.

- `-bits {512|1024|2048|4096}`

  Specify the bit length of the Web server private key.

(3) Usage example

To create the `httpsdkey.pem` Web server private key:

```
keygen -rand file1:file2:file3:file4:file5 -out httpsdkey.pem -bits 1024
```

`file1`, `file2`, `file3`, `file4` and `file5`: Arbitrary files

5.2.2 Creating a Certificate Signing Request (CSR) (certutil reqgen command)

This subsection describes how to use the `certutil reqgen` command to create a Certificate Signing Request (CSR). The created CSR file is submitted to the CA, which then issues the signed certificate. The CSR is created in the format conforming to PKCS #10.

(1) Format

```
certutil reqgen [-sign {MD5|SHA1|SHA224|SHA256|SHA384|SHA512}] -key key-file -out CSR-file
```

(2) Parameters

- `[-sign {MD5|SHA1|SHA224|SHA256|SHA384|SHA512}]`

  Specify the signature algorithm used when the CSR is created.

  `-MD5`: `md5WithRSAEncryption` is used.

  `-SHA1`: `sha1WithRSAEncryption` is used.

  `-SHA224`: `sha224WithRSAEncryption` is used.

  `-SHA256`: `sha256WithRSAEncryption` is used.

  `-SHA384`: `sha384WithRSAEncryption` is used.

  `-SHA512`: `sha512WithRSAEncryption` is used.

- `-key` *key-file*

  Specify the Web server private key file.

- `-out` *CSR-file*

  Specify the file to which the created CSR is output.

(3) Use example

To create the `httpsdkey.pem` Web server private key:

```
certutil reqgen -sign SHA1 -key httpsdkey.pem -out httpsd.csr
```

`httpsdkey.pem`: Key file

`httpsd.csr`: CSR file

5.2.3 Displaying the contents of a Certificate Signing Request (CSR) (certutil req command)

This subsection explains how to display the contents of a Certificate Signing Request (CSR).

(1) Format

```
certutil req -in CSR-file -text
```

(2) Parameter

- `-in` *CSR-file*

  Specify the CSR file to be displayed.

(3) Usage example

```
certutil req -in httpsd.csr -text
```

`httpsd.csr`: CSR file to be displayed

5.2.4 Displaying certificate contents (certutil cert command)

This subsection explains how to display the contents of a certificate file.

The following command displays the part of the certificate file that begins with `-----BEGIN CERTIFICATE-----` and ends with `-----END CERTIFICATE----`.

(1) Format

```
certutil cert -in certificate-file -text
```

(2) Parameter

- `-in` *certificate file*

  Specify the certificate file to be displayed.

(3) Usage Example

```
certutil cert -in httpsd.pem -text
```

`httpsd.pem`: Certificate file to be displayed

5.2.5 Converting the certificate format (certutil cert command)

This subsection explains how to convert the certificate format. Use this functionality as necessary.

(1) Format

```
certutil cert -inform input-format -outform output-format -in input-file -out output-file
```

(2) Parameters

- `-inform` *input-format*

  Input format: {`DER`|`PEM`}
- `-outform` *output-format*

  Output format: {`DER`|`PEM`}
- `-in` *input-file*

  Specify the certificate file before conversion.
- `-out` *output-file*

  Specify the certificate file after conversion.

5.2.6 Creating a hash link (in UNIX) (certutil cert command)

To perform a certificate validity check, specify the certificate of the certificate issuer CA in the SSLCACertificateFile directive or SSLCACertificatePath directive. In the SSLCACertificatePath directive, specify the directory that stores the symbolic link (hash link) with the hash value that points to the certificate of the certificate issuer CA. The hash value is created by using the `certutil cert` command.

If the SSLCACertificatePath directive is specified, the certificate search can be performed efficiently on the Web server by using the hash value. If there are many CA certificates, we recommend that you specify the SSLCACertificatePath directive rather than the SSLCACertificateFile directive. Note that one hash value must be assigned per certificate, so you cannot specify a file with multiple certificates when creating the hash link.

When generating the symbolic link in the hash link directory that is specified in the SSLCACertificatePath directive, you must add `.0` to the hash value. Grant the read and execution permissions to the directory to be specified in the SSLCACertificatePath directive so that the user specified in the User and Group directives can access the directory.

(1) Format

```
certutil cert -noout -hash -in CA-certificate-file
```

(2) Parameter

- *CA-certificate-file*

  Specify the CA certificate file for which the hash link value is created.

(3) Usage example

An example of the hash link directory and CA certificate for the following directory and file is given below:

`/opt/hitachi/httpsd/conf/ssl/cacerts`: Hash link directory

`/opt/hitachi/httpsd/conf/ssl/cacert/cacert.pem`: Certificate of the CA

```
cd /opt/hitachi/httpsd/conf/ssl/cacerts
ln -s /opt/hitachi/httpsd/conf/ssl/cacert/cacert.pem `certutil cert -noout -hash  -in
/opt/hitachi/httpsd/conf/ssl/cacert/cacert.pem`.0
```

This creates the *xxxxxxxx*`.0` hash link for `/opt/hitachi/httpsd/conf/ssl/cacert/cacert.pem`.

5.2.7 Usage examples of the keygen and certutil commands

This subsection provides examples of how to use the `keygen` and `certutil` commands. The information provided in the following examples, such as for the Common Name item, is fictitious, and any connection with real individuals is purely coincidental.

(1) Generating a private key (keygen command)

The following example shows how to use the `keygen` command to generate a private key.

Usage example

```
# keygen -rand file -des3 -out httpsdkey.pem -bits 1024
Adding 'entropy' into random generator
372 random bytes loaded
RSA Key Generation e= 65537, bits= 1024, primes= 2
Enter PEM passphrase:                         <--- Enter a password.
Verifying password - Enter PEM passphrase:    <--- Re-enter the password.
Key generation successful

#
```

Contents of the private key

The contents of the private key are as follows:

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,0150E8E9D7CFAD87

3iQQl5s4ZSjup+XdoHwXj1lyERYFLczMMt5HNbp0+NPFz8f9iRSeIHZAPZIFG4DM
ajlmwoH5ovB7GbCUXhnb9/sj6biD38abyqON3XDoXB0/h6RTNx4bfIR6H0/3EMsF
dNZzC5g2kiHppSuToVe8m6bAM+XNqt6Fq684muZ7t9m4/gVk0nm7ac/OdUuqTN5v
DUE2VPav9OnPd+ZM4s4W31SUs+3b1/UJ2UXFmJFwd6SJ58IKa/d179tHTHndawAE
T1yHEOpAiOOjFj8v3FzHGtqnGMU4dBHsc8PbJIHxJGxPH13X0v69aypzFwi/iNXP
mQmFIxl3KJmPVkuXeNksIkELGJiWuXIJsKVueIWxDoQjHRezOl0nvjh74mJkknSB
GXJMdZfC2IJkHRpMqiliT84WlFJaSobtRvtAfTi/Zq3g/kjvIs451HAQ3m3AUwaa
JjbAqOa2NuTv0URoNKUDKJWK+DQmYLwBLux2hqjpHWxaHMrndqUY8VqU7Nj9SPn3
H7nwu49krgw+/ZG/Y3u+6SRsXLbRUimMtStlKRTm3ECmeFRzm4Csjch/ysGNO0hj
AZxQNdD4tQapNmh164CB+XUEPLjEi3QiT56TYvGzNcSn59vtvY8351eGxXV2++Qf
U8nK7x+6I8Relt1UldDE7h2jHqPWgdHxrzlR6/fwZK4kamQH8qSUfeJr8uVyyr85
izpFPX1/jwueEFLWVmoGu95gkxVRLUotFMWUc+z5FTiUmfDGD/IgCIUB/v54nOWh
X7YVUeWz8W2WZst2P4qm4v71DOEeWRgPIgS1jCVRI6WUTPKGPi+USQ==
-----END RSA PRIVATE KEY-----
```

(2) Creating a Certificate Signing Request (CSR) (certutil command)

This subsection explains how to use the `certutil reqgen` command to create a Certificate Signing Request (CSR). Submit the created CSR file to a CA to receive a signed certificate. Note that if you set a password when creating the private key of the Web server, you are also requested to enter the private key password when creating the CSR.

Specify the items and contents according to the instructions provided by the CA to which the CSR is submitted.

Usage example

```
# certutil reqgen -sign SHA1 -key httpsdkey.pem -out httpsd.csr
Enter PEM passphrase:           <--- Enter the private key password.
Country Name (2 letter code) :JP
State or Province Name (full name) :Kanagawa
Locality Name (eg, city) :Yokohama-shi
Organization Name (eg, company) :HITACHI
Organizational Unit Name (eg, section) :WebSite
Common Name (eg, YOUR name) :www.hws.hitachi.co.jp
Email Address :
Subject:
C=JP,ST=Kanagawa,L=Yokohama-shi,O=HITACHI,OU=WebSite,CN=www.hws.hitachi.co.jp
A certificate signing request was successfully created
#
```

CSR format

The CSR format is as follows:

```
-----BEGIN CERTIFICATE REQUEST-----
MIIBuzCCASQCAQAwezEeMBwGA1UEAxMVd3d3Lmh3cy5oaXRhY2hpLmNvLmpwMRAw
DgYDVQQLEwdXZWJTaXRlMRAwDgYDVQQKEwdISVRBQ0hJMRUwEwYDVQQHEwxZb2tv
aGFtYS1zaGkxETAPBgNVBAgTCEthbmFnYXdhMQswCQYDVQQGEwJKUDCBnzANBgkq
hkiG9w0BAQEFAAOBjQAwgYkCgYEArZZyYumQcY8h4AppAz447H9R+Srzrt08eSzr
yZT8HYrDXz9I8XH6bMMahO4M6u2YI9iVzepQU1uI0f8bCwkFageBWwVQmDwcyJYf
1kY5X+2OgFEYV8CTu7I+A70VlYHobpM/FlBkzUVWD9/fTob0ALYNF9eTbFAL0c6U
sJBZfSsCAwEAAaAAMA0GCSqGSIb3DQEBBQUAA4GBAEiq+yGSVblaOuljyrAei9r3
n5mXtE5KXzQRz0cy6N5BaEV0l9KOtUaTCallsZmQdZ/6dZRSaE27xf/2UF3UxlCC
0+qrG10iQgDe5huSsqBnGGghJB2OPVUJh5S7YC6Ub6HRdOzq7H+D0qvsBC2C0dA/
cCkp8UsRzIjlDW8SVBZO
-----END CERTIFICATE REQUEST-----
```

5.3 Operating CRL

5.3.1 Downloading CRL

Access the LDAP server and download the invalid certificate list (CRL) from the specified attribute. Confirm in advance, the entry, attribute, and the CRL format to acquire the CRL with the LDAP administrator.

The crldownload command that downloads the CRL is described below:

The crldownload command cannot be used in the HP-UX (IPF) version of the product.

(1) Format

```
crldownload -b search-base-DN -L LDAP-library-name -o file-name[-a attribute][-D
bind-DN][-h host-name][-H][-p port-number][-w password]
```

(2) Option

- -b *search-base-DN*

  Specify DN of the stored entry of the CRL.
- -L *LDAP-library-name*

  Specify the LDAP library file name to be used.
- -o *file-name*

  Specify the file name that contains the CRL output.
- -a *file-name* ~<<certificateRevocationList;binary>>

  Specify the attribute that stores the CRL.
- -D *bind-DN*

  Specify the DN to bind. If you omit this option, an anonymous bind is executed.
- -h *host-name* ~<<localhost>>

  Specify the host name or the IP address of the LDAP server to be accessed.
- -H

  Specify when help is to be displayed. You cannot use this option in combination with other options.
- -p *port-number* ~((1-65535))<<389>>

  Specify the port number of the LDAP server to be accessed.

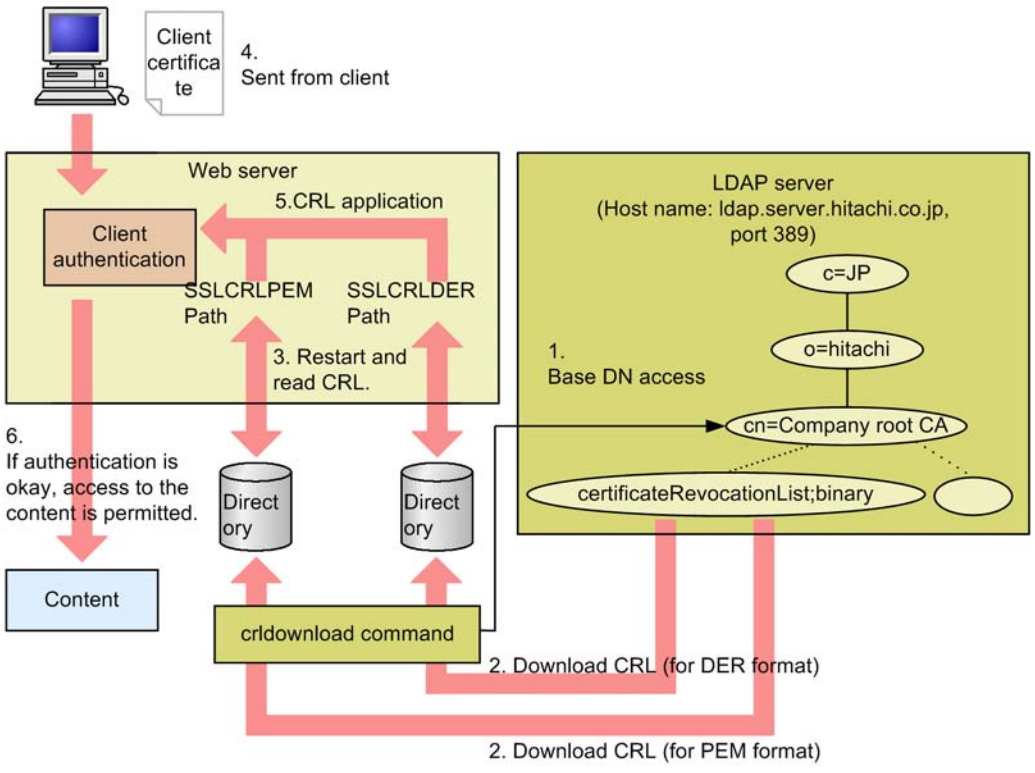
- -w *password* ~<<NULL>>

  Specify a password to be used when binding to the bind DN. When you omit this option, the password is not used.

(3) How to use

The following figure shows how to use the crldownload command.

Figure 5-3: Howto use the crldownload command



1. Use the crldownload command to access the entry storing the CRL.
2. Acquire the CRL from the entry attributes. When the stored CRL is in the DER format, download the CRL in the directory specified in the SSLCRLDERPath directive. When the CRL is in PEM format, download the CRL in the directory specified in the SSLCRLPEMPath directive.
3. Restart the Web server.
4. The client sends the certificate when accessing by the SSL.
5. Apply the acquired CRL when authenticating the client certificate.

6. If authentication succeeds, the client can access the contents.

#

If there are files in the inappropriate format in directories specified in the SSLCRLPEMPath and the SSLCRLDERPath directives, the Web server is not started. Therefore, when you use the crldownload tool, make sure that the CRL is in the correct format before you store the CRL in these directories.

(4) Usage example

(a) Downloading CRL in the DER format

Execute the following script to download the CRL. The script file name is `/opt/hitachi/httpsd/sbin/hws_getCRL.sh` in UNIX version, and `<`*`Cosminexus-installation-directory`*`>\httpsd\sbin\hws_getCRL.bat` in Windows version.

- Executable script contents

  Download the CRL that is stored in the DER format on the LDAP server and restart the Web server. At that time, messages that are output are stored in a log file.

  The CRL storage destination File name

    UNIX Version

    /opt/hitachi/httpsd/conf/ssl/crl/DER/rootCA.crl

    Windows Version

    <*Cosminexus-installation-directory*>\httpsd\conf\ssl\crl\DER\rootCA.crl

  Log file name

    UNIX Version

    /opt/hitachi/httpsd/conf/ssl/crl/crl.log

    Windows Version

    <*Cosminexus-installation-directory*>\httpsd\conf\ssl\crl\crl.log

Script contents (for UNIX Version)

```
#!/bin/sh
#parameters
LIB=LDAP library name
HOST="ldap.server.hitachi.co.jp"
PORT="389"
BASE="cn=Company root CA, o=Hitachi, c=JP"
ATTR="certificateRevocationList;binary"
FILE="/opt/hitachi/httpsd/conf/ssl/crl/DER/rootCA.crl"

LOG="/opt/hitachi/httpsd/conf/ssl/crl/crl.log"
TMP="/opt/hitachi/httpsd/conf/ssl/crl/tmp-rootCA.crl"

#download
TOOL="/opt/hitachi/httpsd/sbin/crldownload"
HTTPSD="/opt/hitachi/httpsd/sbin/httpsdctl graceful"
LOGTIME=""
LANG=C

if `$TOOL -L $LIB -h $HOST -p $PORT -b "$BASE" -a "$ATTR" -o $TMP >> $LOG
2>&1`then
  if `mv -f $TMP $FILE >> $LOG 2>&1`
  then
    $HTTPSD >> $LOG 2>&1
    exit 0
  else
    LOGTIME=`date`
    echo "[$LOGTIME] Moving $TMP to $FILE failed" >> $LOG
    rm -f $TMP >> /dev/null 2>&1
  fi
else
  LOGTIME=`date`
fi
echo "[$LOGTIME] Stop restarting Cosminexus HTTP Server." >> $LOG
exit 1
```

Script contents (for Windows Version)

```
@echo off
REM
REM #parameters
SETLOCAL
SET LIB="<Cosminexus-installation-directory>\httpsd\libldap\nsldap32v50.dll"
SET HOST="ldap.server.hitachi.co.jp"
SET PORT="389"
SET BASE="cn=Company root CA, o=Hitachi, c=JP"
SET ATTR="certificateRevocationList;binary"
SET
FILE="<Cosminexus-installation-directory>\httpsd\conf\ssl\crl\DER\rootCA.cr
l"

SET FORM="DER"
SET LOG="<Cosminexus-installation-directory>\httpsd\conf\ssl\crl\crl.log"
SET
TMPCRL="<Cosminexus-installation-directory>\httpsd\conf\ssl\crl\tmp-rootCA.
crl"

REM #download
SET TOOL="<Cosminexus-installation-directory>\httpsd\sbin\crldownload.exe"
SET HTTPSD="<Cosminexus-installation-directory>\httpsd\httpsd.exe"

%TOOL% -L %LIB% -h %HOST% -p %PORT% -b %BASE% -a %ATTR% -o %TMPCRL% >> %LOG%
2>&1 || GOTO ERR

COPY %TMPCRL% %FILE% >> %LOG% 2>&1 || GOTO CPERR
DEL %TMPCRL% >> %LOG% 2>&1
%HTTPSD% -n "Cosminexus HTTP Server" -k restart >> %LOG% 2>&1
GOTO TOOLEND

:CPERR
 ECHO Moving %TMPCRL% to %FILE% failed >> %LOG%
 DEL %TMPCRL% >> %LOG% 2>&1
 DEL %TMPSSL% >> %LOG% 2>&1
 GOTO ERR

:ERR
 ECHO Stop restarting Cosminexus HTTP Server. >> %LOG%

:TOOLEND
 endlocal
 echo on
```

■ How to execute the script (for UNIX Version)

```
/opt/hitachi/httpsd/sbin/hws_getCRL.sh
```

■ How to execute the script (for Windows Version)

```
C:\> "<Cosminexus-installation-directory>\httpsd\sbin\hws_getCRL.bat"
```

(b) Downloading CRL and restarting the Web server at regular intervals

■ UNIX Version

A super user or a user with permission from the super user can periodically download the CRL using the `crontab` command# and can restart the Web server.

In the `crontab` command, specify the time to execute the script that describes the `crldownload` or the `crldownload` command, download the CRL periodically and restart the Web server.

# This command is an OS command. Use this command to register and control the job to the cron. For details on how to specify the command, see the OS manual.

How to specify the `crontab` command

```
# crontab -
minute hour date month day command
```

Each `crontab` file entry is configured with a line that consists of six fields. Each field is delimited with a space or a tab, and each line includes the respective value as shown below:

Minutes: Minutes for command execution (from 0 to 59).

Hours: Hours for command execution (from 0 to 23).

Date: Date for command execution (from 1 to 31).

Month: Month for command execution (from 1 to 12).

Days: Days for command execution (from 0 to 6, to specify Monday to Saturday).

Command: Shell commands for execution.

* (Asterisk) denotes all valid values.

An example of the `crontab` command

Specify the following command to download the CRL every day at 8 a.m. and restart the Web server (to execute the script mentioned in (a)):

```
# crontab -
0 8 * * * /opt/hitachi/httpsd/sbin/hws_getCRL.sh
(stop the entry with <Ctrl>+<d> key)
#
```

- Windows Version

You can use the `at` command# to periodically download the CRL and to restart the Web server.

In the `at` command, specify the time to execute a script that code the `crldownload` or `crldownload` command to download the CRL periodically and restart the Web server.

# This command is an OS command. For details on how to specify the command, see the OS manual.

How to specify the `at` command

```
C:\>at [\\Computer name]Time[/every:Date[,...]|/next:Date[,...]]"Command"
```

An Example of the `at` command

Specify the following command to download the CRL everyday at 8 a.m. and restart the Web server (to execute the script mentioned in (a)):

```
C:\>at 8:00 /every:M,T,W,Th,F,S,Su
"<Cosminexus-installation-directory>\httpsd\sbin\hws_getCRL.bat"
```

5.4 Using the server private key with a password

When you use the server private key protected by a password, you can save the password in advance in a file and set the directive to omit the password input when restarting the server. This procedure is described below. Note that the following procedure is mandatory when you use the server private key protected by password in Windows version of Cosminexus HTTP Server:

1. Create a server private key with a password by using the keygen command.
2. Create a password file by the sslpasswd command.
3. Set the SSLCertificateKeyPassword directive that specifies the created password file together with the SSLCertificateKeyFile directive that specifies the server private key file in the `httpsd.conf`.
4. Start or restart the server.

You need to take care when protecting the password file items. Set the directory permissions and the file permissions to prevent other users from accessing the storage directory of the server private key, and also prevent them from accessing the storage directory of the password file.

The following section describes the sslpasswd command that creates the password file that is to be specified in the SSLCertificateKeyPassword directive.

5.4.1 sslpasswd command

(1) Format

```
sslpasswd server-private-key-file-name password-file-name
```

(2) Parameters

- *server-private-key-file-name*

  Specify the password protected server private key.

- *password-file-name*

  Specify the name of the file that outputs password.

(3) Usage example

```
sslpasswd httpsdkey.pem .keypasswd
```

(4) Notes

- You cannot specify an existing file name as the password file name.
- You cannot use the password file created with the sslpasswd command on Windows in UNIX version.
- You cannot use the password file created with the sslpasswd command on UNIX in Windows version.

Chapter

6. Directives

This chapter describes the directives that are defined in the configuration files (the `httpsd.conf` file and the access control file).

6.1 List of directives

6.1.1 List of directives

At a minimum, the directives below must be set to start Cosminexus HTTP Server.

- Minimum directives to be set:

  User (UNIX version)

  Group (UNIX version)

  ServerName

  SSLDisable (when not using SSL)

- Additional minimum directives to be set when using SSL:

  SSLCertificateFile

  SSLCertificateKeyFile

The table below lists the directives that you can specify in the configuration file.

The next two symbols are used in the table below and in the descriptions of directives. Ⓤ : This symbol denotes the directive that is valid only in the UNIX version.

Ⓦ : This symbol denotes the directive that is valid only in the Windows version.

Table 6-1: Directive list

Settings	Directives	Multiple specification
Definitions of blocks in httpsd.conf file	<Directory>	Y
	<DirectoryMatch>	Y
	<Files>	Y
	<FilesMatch>	Y
	<IfModule>	Y
	<Limit>	Y
	<Location>	Y
	<LocationMatch>	Y

Settings	Directives	Multiple specification
	<VirtualHost>	Y
Basic definition of server	ServerName	N
	Port	N
	User Ⓤ	N
	Group Ⓤ	N
	ServerAdmin	N
	ServerRoot	N
	ServerSignature	N
	Listen	Y
	BindAddress	N
	LoadModule	Y
	LoadFile	Y
	Include	Y
	ExtendedStatus	N
	ServerTokens	N
	CoreDumpDirectory Ⓤ	N
	FileETag	Y
Definitions for managing the contents	UserDir	Y
	DocumentRoot	N
	ErrorDocument	Y
Definitions for the requests from Web browser (Alias)	Alias	Y
	AliasMatch	Y
	Redirect	Y
	RedirectMatch	Y

Settings	Directives	Multiple specification
Definition of responses to Web browser	HWSNotModifiedResponseHeaders	Y
Definitions for MIME types	DefaultType	N
	TypesConfig	N
	AddCharset	Y
	AddDefaultCharset	N
	AddType	Y
	ForceType	N
	HWSErrorDocumentMETACharset Ⓦ	N
Definitions for content negotiation	LanguagePriority	Y
	AddEncoding	Y
	AddLanguage	Y
	DefaultLanguage	N
	CacheNegotiatedDocs	N
	MultiviewsMatch	N
Definitions for handler	AddHandler	Y
	SetHandler	N
Definitions for Web server performance	StartServers Ⓤ	N
	MinSpareServers Ⓤ	N
	MaxSpareServers Ⓤ	N
	MaxClients Ⓤ	N
	MaxRequestsPerChild Ⓤ	N
	Timeout	N

Settings	Directives	Multiple specification
	ListenBacklog	N
	ThreadsPerChild Ⓦ	N
	HWSMaxQueueSize Ⓦ	N
	HWSKeepStartServers Ⓤ	N
Definitions for KeepAlive	KeepAlive	N
	MaxKeepAliveRequests	N
	KeepAliveTimeout	N
Definitions to limit requests	LimitRequestBody	N
	LimitRequestFields	N
	LimitRequestFieldsize	N
	LimitRequestLine	N
Definitions of CGI and environment variables	ScriptAlias	Y
	ScriptAliasMatch	Y
	UseCanonicalName	N
	BrowserMatch	Y
	BrowserMatchNoCase	Y
	PassEnv	Y
	SetEnv	Y
	UnsetEnv	Y
	SetEnvIf	Y
	SetEnvIfNoCase	Y
	Action	Y
	Script	Y
	ScriptInterpreterSource Ⓦ	N

Settings	Directives	Multiple specification
	HWSSetEnvIfIPv6	Y
Definitions for the display contents of directory index	DirectoryIndex	N
	FancyIndexing	N
	AddIconByEncoding	Y
	AddIconByType	Y
	AddIcon	Y
	DefaultIcon	N
	ReadmeName	N
	HeaderName	N
	IndexIgnore	Y
	IndexOrderDefault	N
	AddAltByEncoding	Y
	AddAltByType	Y
	AddAlt	Y
	AddDescription	Y
	IndexOptions	Y
Definitions to control Web server access	AccessFileName	N
	AllowOverride	N
	AuthName	N
	AuthType	N
	AuthGroupFile	N
	AuthUserFile	N
	AuthAuthoritative	N
	Require	Y
	Options	N

Settings	Directives	Multiple specification
	Order	N
	Allow from	Y
	Deny from	Y
	Satisfy	N
	TraceEnable	N
	IdentityCheck Ⓤ	N
Definitions for the SSL encryption and authentication	SSLRequireSSL	N
	SSLEnable	N
	SSLDisable	N
	SSLCertificateFile	N
	SSLCertificateKeyFile	N
	SSLCACertificatePath Ⓤ	N
	SSLCACertificateFile	N
	SSLVerifyClient	N
	SSLVerifyDepth	N
	SSLRequiredCiphers	N
	SSLRequireCipher	Y
	SSLBanCipher	Y
	SSLDenySSL	N
	SSLFakeBasicAuth	N
	SSLCacheServerPort Ⓤ	N
	SSLSessionCacheTimeout	N
	SSLCacheServerPath Ⓤ	N

Settings	Directives	Multiple specification
	SSLCacheServerRunDir Ⓤ	N
	SSLSessionCacheSize	N
	SSLSessionCacheSizePerChild Ⓤ	N
	SSLCRLAuthoritative	N
	SSLCRLDERPath	N
	SSLCRLPEMPath	N
	SSLExportCertChainDepth	N
	SSLExportClientCertificates	N
	SSLCertificateKeyPassword	N
	SSLProtocol	Y
Definitions to show the Web server in multiple hosts according to the operation status	NameVirtualHost	Y
	ServerAlias	Y
	ServerPath	N
Definitions for the image map file	ImapDefault	N
	ImapBase	N
	ImapMenu	N
	HWSImapMenuCharset	N
Definitions for the logs to be collected	HostnameLookups	N
	ErrorLog	N
	LogLevel	N
	LogFormat	Y
	CustomLog	Y
	TransferLog	Y

Settings	Directives	Multiple specification
	PidFile	N
	ScriptLog	N
	ScriptLogBuffer	N
	ScriptLogLength	N
	HWSLogSSLVerbose	N
	HWSLogTimeVerbose	N
	HWSRequestLog	N
	HWSRequestLogType	N
	HWSSuppressModuleTrace	Y
	HWSErrorLogClientAddr Ⓤ #	N
Definitions for the directory servers	LDAPBaseDN	N
	LDAPRequire	Y
	LDAPServerName	N
	LDAPServerPort	N
	LDAPSetEnv	Y
	LDAPTimeout	N
	LDAPUnsetEnv	Y
	LDAPNoEntryStatus	N
Definitions for the trace to be collected	HWSTraceIdFile	N
	HWSTraceLogFile	N
	HWSStackTrace Ⓤ	N
Definitions for the reverse proxy	ProxyPass	Y
	ProxyPassReverse	Y
	ProxyVia	N

Settings	Directives	Multiple specification
	ProxyErrorOverride	N
	ProxyPreserveHost	N
	HWSProxyPassReverseCookie	Y
Definition for the flow restricting functionality	QOSCookieDomain	N
	QOSCookieExpires	N
	QOSCookieName	N
	QOSCookieSecure	N
	QOSCookieServers	N
	QOSRedirect	Y
	QOSRejectionServers	N
	QOSResponse	N
Definitions for the header customization functionality	Header	Y
	RequestHeader	Y
Definitions for the validity period setting functionality	ExpiresActive	N
	ExpiresByType	Y
	ExpiresDefault	N
Definitions for the planned termination	HWSGracefulStopLog	N
	HWSGracefulStopTimeout	N
Definitions for the static contents cache functionality	HWSContentCacheMaxFileSize Ⓦ	N
	HWSContentCacheSize Ⓦ	N

Legend:

Y: Can be specified.

N: Cannot be specified.

Note:

Unless it is specifically mentioned, you can describe the file name in a format that includes the directory name (with path information).

#

You can use this functionality only in the HP-UX (IPF) version of the product.

6.1.2 Descriptive conventions for directives

(1) Regular expressions

The following table describes the regular expressions that you can use to specify directives:

Table 6-2: Regular expressions

Symbol	Functionality	Usage example	Meaning of the usage example
.	Indicates one optional character.	a...c	Three optional characters and then character c follow character a. matches with abcdc.
*	Indicates that the preceding character is repeated for zero or more times.	ab*cd*	Matches with ac, abbbbc, and abbbbcd.
+	Indicates that the previous character is repeated for one or more times.	ab*c+	Matches with abbbc. Does not match with abbb.
?	Is there a previous character?	abbbc?	Matches with abbbc and abbb.
|	Denotes the OR operation	a|bc|d	Matches with a, bc, or d.
¥	A prefix for a special character (. ^$*+?| ¥ []()\{\}). Three ¥ symbols are used to express a single ¥¥¥ .	¥ .	Matches with a "." (period).
		¥¥¥	Matches with a single ¥ character.
^	Matches at the beginning of a line.	^ab	Matches with abcde.
$	Matches at the end of a line.	abc$	Matches with aaabc.

Symbol	Functionality	Usage example	Meaning of the usage example
{m}	Indicates repetition of the preceding regular expressions for m number of times.	a{5}	Matches with aaaaa.
{m,}	Indicates repetition of the preceding regular expressions for minimum m number of times.	a{3,}	Matches with aaa and aaaa. Does not match with aa.
{m,n}	Repetition of the preceding regular expressions for minimum m and maximum n number of times.	a{3,5}	Matches with aaa, aaaa, and aaaaa. Does not match with aa and aaaaaa.
[Character string]	Indicates one of the optional characters# included in the character string.	[abc]* or [a-c]*	Matches with aaa, bbb, ccc, cba, and aab.
[^Character string]	Indicates one of the optional characters that are not included in the character string.	[^0-9]	Matches with one non-numeric character.
(Character string)	Group the character strings.	(ab)+	Matches with ababab. Does not Match with ababb.
		aa(xx|yy)bb	Matches with aaxxbb and aayybb.

#

The following three characters have a specific meaning in the [character string].

^: It is specified after [and used to indicate the characters that are not included in the character string.

]: It is used to denote the end of the character string.

-: It is used to denote the range of characters.

Note that the ¥ that comes before these special character is omitted.

Specify the characters having special meaning in the [character string] as normal characters by following method. Note that, special characters excluding ^] - \ are interpreted as normal characters.

^ : Do not specify at the beginning of the character string. (Example)[ab^yz]

]: Specify at the beginning of the character string. (Example)[]abxy]

-: Specify at the end. (Example)[abxy-]

¥ : Specify as ¥¥¥ . (Example)[¥¥¥ abxy]

(2) Path information to be specified in directives

In the case of directives that specify directory names, file names, or the path names, the path information that you can specify differs based on the directive type.

The path types are as shown below. Path information of each directive is to be described in the respective directive.

- You can specify only absolute paths. (For the Windows version, the drive name is also included in the absolute path.)
- You can specify path information using a relative path only when using the ServerRoot directive (and the ServerRoot directive must be specified in advance.)

Note that directories and files on the network cannot be specified as path information. Directories and files on a file system that uses the network also cannot be specified.

(3) Comment lines

In the configuration file, if you add a hash mark (#) to the beginning of a line, that line becomes a comment line. However, if you enter a character string that begins with a hash mark (#) after a specified directive, the character string after the hash mark (#) is not commented out. Examples of specifying comment lines are as follows:

Corret example

```
#Deny from all
```

This line is a comment line.

Incorrect example

```
Deny from all    #comment
```

`#comment` is a directive specification value. `#comment` is not a comment.

(4) Notes on specifying IPv6 addresses

When an IPv6 address is specified in a directive, enclose the IPv6 address in square brackets ([ ]), as in [*IPv6-address*]. When an IPv6 address and a port number are specified in a directive, enclose the IPv6 address in square brackets ([ ]) and specify the port number after a colon (:), as in [*IPv6-address*]:*port-number*.

However, if IPv6 addresses are specified in the following directives, specify the IPv6 addresses without square brackets ([ ]):

- Allow from directive
- Deny from directive
- HWSSetEnvIfIPv6 directive

Specify a global unicast address when using an IPv6 address.

6.1.3 Descriptive format for directives

This section provides notes regarding the items in *6.2 Directive details*.

(1) Directive names

The following table shows directive names and the specification format that you can use.

Symbols	Meanings
[ ]	You can omit the items inside [ ]. **(Example)** In the case of A[,B][,C], the following four specifications are possible: A A,B A,B,C A,C
{ }	You can specify any one of the items enclosed by { }. **(Example)** In the case of A{,B | ,C}, the following two specifications are apossible: A,B A,C
|	This symbol denotes the demarcation of the selected option.
_ (underline)	This symbol denotes the value that is assumed by the system when you omit the specification of the items.
...	The item immediately preceding this symbol can be repeated.
~	The item immediately preceding this symbol is described according to the syntax rule after this symbol.
<< >>	This symbol denotes the value that is assumed by the system when you omit the item specification.
(( ))	This symbol denotes the range of the value that you can specify.

(2) Location where you can code

The locations where you can code are restricted as per the directives. In section *6.2 Details of the directives*, the location where you can code each directive is mentioned in the following format:

Location where you can code directives	Explanation
httpsd.conf	httpsd.conf files other than the VirtualHost block and Directory block.
<VirtualHost>	VirtualHost block of httpsd.conf file.

Location where you can code directives	Explanation
<Directory>	Directory block, Location block, and Files block of httpsd.conf file
.htaccess	Access control file specified in the AccessFileName directive.
<Location>	Location block of the httpsd.conf file.

Directives are referred to in the following sequence:

1. httpsd.conf file other than the VirtualHost block and Directory block.
2. VirtualHost block of the httpsd.conf file.
3. Directory block of the httpsd.conf file.
4. Access control file.
5. Files block of the httpsd.conf file.
6. Location block of the httpsd.conf file.

You can enable or disable directives defined in the access control file as per the definition of the AllowOverride directive (Overwrite permission level) of the Directory block.

(3) Overwrite permission

Define the permission level when you grant overwrite permission with the AllowOverride directives. Describe the overwrite permission level of each directive in the respective directives. There are multiple permission levels. For more details, see *AllowOverride directive*. You can specify `.htaccess` in the explanation of each directive, and if you do not describe the overwrite permission level, the permission level is 'All'.

6.2 Details of the directives

6.2.1 Directives starting with <

This section explains the directives of block definition in the order of reference:

1. <Directory> directive, <DirectoryMatch> directive, and access control file.
2. <Files> directive and <FilesMatch> directive
3. <Location> directive

(1) <Directory directory-name>...</Directory>

(a) Contents

Specify the <Directory> directive when defining the directive for a specific directory. You can specify the name of the directory in the *directory-name* and specify the blocks that define directives valid only for that directory and its sub-directories.

Specify the directory name with an absolute path.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
<Directory />                                              ...1.
     Options None                                          ...2.
     AllowOverride None                                    ...3.
</Directory>                                               ...4.

<Directory "<Cosminexus-installation-directory>/httpsd/
htdocs">  ...5.
     Options Indexes                                       ...6.
     AllowOverride None                                    ...7.
     Order allow,deny                                      ...8.
     Allow from all                                        ...9.
</Directory>                                               ...10.
```

1. Definition of the root directory
2. Disable all the functions
3. Prohibit all overwriting
4. End of definition
5. Definition of <*Cosminexus-installation-directory*>/httpsd/htdocs directory
6. Directory index can be displayed

7. Prohibit all overwriting
8. Evaluate the Allow directive specification before the Deny directive specification
9. Permit access from all the hosts
10. End of definition

(2) <DirectoryMatch regular-expression>...</DirectoryMatch>

(a) Contents

Specify the <DirectoryMatch> directive when defining directives for a directory that satisfies the conditions described in the regular expressions. You can specify the directory name with regular expressions, and specify the blocks that define directives valid only for that directory and its sub-directories.

Specify the directory name in the regular expressions using an absolute path.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(3) <Files file-name>...</Files>

(a) Contents

Specify <Files> directive when defining directives for specific files. You can specify the name of files in the file name and specify the blocks that define directives valid only for that file.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(4) <FilesMatch regular-expression>...</FilesMatch>

(a) Contents

Specify the <FilesMatch> directive when defining directives for files that satisfy the condition described in the regular expressions. You can specify the file name with regular expressions and specify the blocks that define the directives valid only for that file.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(5) <IfModule [!]module-name>...</IfModule>

(a) Contents

If the specified module is embedded, the directives specified in the block are enabled. If "!" (exclamation mark) is added before the module name, and if the specified module is not embedded, the directives specified in the block are enabled.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(6) Limit method-name [method-name ...]>...</Limit>

(a) Contents

Specify the <Limit> directive when defining directives for a specific HTTP protocol method. You can specify the blocks that define the access control directives valid only for the methods specified in method name. You can specify multiple method names.

method-name: GET, POST, PUT, DELETE, CONNECT, OPTIONS

(HEAD is included in GET)

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Specification example

```
<Directory />
  <Limit PUT DELETE>                     ...1.
    Order deny,allow                     ...2.
    Deny from all                        ...3.
    Allow from .your_domain.com          ...4.
  </Limit>                               ...5.
</Directory>
```

1. Definitions for the `PUT` and the `DELETE` methods
2. Evaluate the specification of the Deny directive before the specification of the Allow directive
3. Deny access to the `PUT` and the `DELETE` methods from all hosts
4. Permit access to the `PUT` and the `DELETE` methods from .your_domain.com
5. End of definition

(7) <Location URL>...</Location>

(a) Contents

Specify the <Location> directive when defining the directives for the request to the locations described in the specific URL. However, you cannot specify the characters from ? (query string) onwards in the URL.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
  <Location /server-status>              ...1.
```

```
     SetHandler server-status         ...2.
     Order deny,allow                 ...3.
     Deny from all                    ...4.
     Allow from .your_domain.com      ...5.
</Location>                           ...6.
```

1. Definition of the URL/server-status
2. Request of this directory is related to the server-status handler
3. Evaluate the specification of the Deny directive before the specification of the Allow directive
4. Deny access from all the hosts
5. Permit access from .your_domain.com
6. End of definition

(8) <LocationMatch regular-expression>...</LocationMatch>

(a) Contents

Specify the <LocationMatch> directive when defining the directives for the request to the URL that satisfies the conditions described in the regular expressions. However, you cannot specify the characters from ? (query string) onwards in the URL.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(9) <VirtualHost {host-name | IP-address[:port-number]} [{host-name | IP-address[:port-number]} ...]>...</VirtualHost>

(a) Contents

Specify the <VirtualHost> directive when defining the directives for the request to the host described in host name or the IP address [:Port number].

Note that the host names corresponding to IPv6 addresses can also be specified. When specifying an IPv6 address for *IP-address*, enclose the IPv6 address in square brackets ([ ]).

(b) Location where you can code

httpsd.conf

(c) Specification example

```
<VirtualHost 172.17.40.30:80>
    :
</VirtualHost>
<VirtualHost [2001::123:4567:89ab:cdef]:80>
    :
```

```
</VirtualHost>
```

6.2.2 Directives starting with A

(1) AccessFileName file-name [file-name ...]

~<<.htaccess>>

(a) Contents

The AccessFileName directive defines the name of the file (access control file) that defines the access control directive. If AllowOverride directive permits the access, this file is referred to every time when the contents are requested and the access control is checked.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
AccessFileName .htaccess
```

The file name of the access control file is .htaccess.

(2) Action {MIME-type | Handler} CGI-script-name

(a) Contents

This directive specifies the executable script with the CGI script name when the Web browser requests the contents specified in the MIME type or the handler. Specify the CGI script name in the URL. When performing multiple specifications of this directive, you cannot specify different CGI scripts with the same MIME type.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
Action image/gif /cgi-bin/images.cgi
```

(3) AddAlt "character-string" extension [extension ...]

(a) Contents

When the directory index is displayed, specify the AddAlt directive for displaying the character string related to a file specified by the extension. You can specify multiple extensions for one character string. You can specify this directive for displaying file attributes in environments such as the text mode Web browser where you cannot display icons.

You can specify the following in the extension:

- File extension
- File name or file extension with wild card notation
- File name

When performing multiple specifications of this directive, you cannot specify different character strings with the same extension.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
AddAlt "HTML" htm html
```

When the file extension is htm or html, the character string "HTML" is displayed.

(4) AddAltByEncoding "character-string" MIME-encoding [MIME-encoding ...]

(a) Contents

When the directory index is displayed, specify the AddAltByEncoding directive for displaying the character strings related to the MIME encoding (such as x-compress) in an environment where you cannot display icons. You can specify multiple MIME encoding for one character string. When performing multiple specifications of this directive, you cannot specify different character strings with the same MIME type.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
AddAltByEncoding "gzip" x-gzip
```

(5) AddAltByType "character-string" MIME-type [MIME-type ...]

(a) Contents

When the directory index is displayed, specify the AddAltByType directive for displaying the character string related to the MIME type (such as text/html) in an environment where you cannot display icons. You can specify multiple MIME types for one character string. When performing multiple specifications of this directive, you cannot specify different character strings with the same MIME type.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
AddAltByType "plain text" text/plain
```

(6) AddCharset character-set extension [extension ...]

(a) Contents

The AddCharset directive specifies the character sets for file extension. The character set is set as the value of charset= in the Content-Type header. Use this directive for clearly specifying the character sets for the client. When performing multiple specifications of this directive, you cannot specify different character strings with the same extension. Specify the file extensions specified in the AddType directive or the TypesConfig directive, and file extension must be related to MIME types

(b) Location where you can code

httpsd.conf, <VirtualHost>, <directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
AddCharset EUC-JP .euc
AddCharset ISO-2022-JP .jis
AddCharset SHIFT_JIS .sjis
```

(7) AddDefaultCharset [On | Off | character-set]

(a) Contents

The AddDefaultCharset directive specifies the default value of the character set for a file extension. The specified default value becomes the default value for the settings of the AddCharset directive. The Content-Type is set as a default for text/plain and text/html.

On: ISO-8859-1 is set as the default character set.

Off: The character set is not set.

character-set: The specified character set is used as the default character set.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
AddDefaultCharset ISO-2022-JP
```

(8) AddDescription "character-string" file-name [file-name ...]

(a) Contents

Specify the AddDescription directive when you want a character string to be displayed as a description for the file extension specified in the file name, the wild card notation file name, or the complete file name without path information when the directory index format is displayed. Note that, when a character string ending with a slash is specified in the file name, the character string is interpreted as a wild card specification with an * added.

You can specify the following in the file name:

- File extension
- File name with the wild card notation
- File name

When performing multiple specifications of this directive, you cannot specify different character strings for the same file name.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
AddDescription "The planet Mars" /web/pics/mars.gif
```

(9) AddEncoding compression-format-extension

(a) Contents

The AddEncoding directive specifies the relationship between the compression format and the extension required when the compressed data in the Web server is displayed on the Web browser. This directive is set when the Web server sends the Content-Encoding header to the Web browser as information of the compressed file deployment. The operation in which this header is used depends upon the implementation of the Web browser. When performing multiple specifications of this directive, you cannot specify different compression formats for the same extension.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
AddEncoding x-compress Z    Compression format of the file with
extension Z is x-compress
AddEncoding x-gzip gz       Compression format of the file with
extension gz is x-gzip
```

(10) AddHandler handler-name extension [extension]

(a) Contents

Define the AddHandler directive to support the file extension processed by the handler.

Handler names that you can specify are as follows. When performing multiple specifications of this directive, you cannot specify different handler names for the same extension.

cgi-script: Execution of CGI script

imap-file: Image map processing

server-status: Status display

hws_cache: Cache of static contents

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
AddHandler cgi-script .cgi          Extension.cgi is cgi-script
handler
AddHandler imap-file map             Extension map is imap-file
handler
```

(11) AddIcon {(character-string, URL) | URL} extension [extension ...]

(a) Contents

Specify the AddIcon directive to support and display the icons of the directory index for the extension. In the character string specify the characters to be displayed when the web browser cannot display the images. Specify the URL of the icon image file in the URL. When specifying the image file on the local host, you can omit 'http: //IP address' of the URL. Note that if you specify IPv6 addresses without omitting http:// *IP-address* of the URLs, enclose each IPv6 address in square brackets ([ ]).

You can specify the following as an extension:

- File extension
- File extension or the file name with wild card notation
- File name

If you code ^^DIRECTORY^^ as the extension, you can set the icons for directories. If ^^BLANKICON^^ is specified as the extension, you can set the icons to match the header indent of the displayed contents when the directory index is displayed.

When performing multiple specifications of this directive, you cannot specify different character strings and URLs for the same extension.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
AddIcon /icons/tar.gif .tar
```

Icon definitions when the extension is .tar

```
AddIcon /icons/layout.gif .html .shtml .htm .pdf
```

Icon definitions when the extension is .html, .shtml, .htm, and .pdf

```
AddIcon /icons/text.gif .txt
```

Icon definitions when the extension is .txt

```
AddIcon /icons/back.gif ..
```

Icon definitions of the parent directory

```
AddIcon /icons/hand.right.gif README
```

Icon definitions of the README file

```
AddIcon /icons/folder.gif ^^DIRECTORY^^
```

Icon definitions for a directory

```
AddIcon /icons/blank.gif ^^BLANKICON^^
```

Indent icon definition of the directory index header

```
AddIcon http://[2001::123:4567:89ab:cdef]/icons/text.gif .txt
```

Icon definitions when an IPv6 address is specified

(12) AddIconByEncoding {(character-string, URL) | URL} MIME-encoding [MIME-encoding ...]

(a) Contents

Specify the AddIconByEncoding directive in the case of displaying the icon with the support of MIME encoding when the directory index format is displayed. In the character string specify the characters to be displayed when the Web browser cannot display the images. Specify the URL of the icon image file in the URL. When specifying the image file on the local host, you can omit 'http://IP address' in the URL. Note that if you specify IPv6 addresses without omitting http://*IP-address* of the URLs, enclose each IPv6 address in square brackets ([ ]).

When performing multiple specifications of this directive, you cannot specify different character strings and URLs for the same MIME type.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip
```

Icon definitions for MIME encoding x-compress and x-gzip

(13) AddIconByType {(character-string, URL) | URL} MIME-type [MIME-type ...]

(a) Contents

Specify the AddIconByType directive in the case of displaying the icon with the support of MIME encoding when the directory index format is displayed. In the character string you can specify the characters to be displayed when the Web browser cannot display the images. You can specify the location of the image file name of the icon to be displayed in the URL. Note that if you specify IPv6 addresses without omitting http://*IP-address* of the URLs, enclose each IPv6 address in square brackets ([ ]).

When performing multiple specifications of this directive, you cannot specify different file names for the same MIME type.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
AddIconByType (TXT,/icons/text.gif) text/*
```

Icon definition for the MIME type text/*

```
AddIconByType (IMG,/icons/image2.gif) image/*
```

Icon definition for the MIME type image/*

```
AddIconByType (SND,/icons/sound2.gif) audio/*
```

Icon definition for the MIME type audio/*

```
AddIconByType (VID,/icons/movie.gif) video/*
```

Icon definition for the MIME type video/*

(14) AddLanguage *language-code extension*

(a) Contents

The AddLanguage directive specifies the language to be used in the document. The language code is set in the Content-Language response header. If you specify this directive, you can perform the content negotiation to select contents that the Web server sends, when the language setting of Web browser sets the priority order (Accept-Language header) of the language code in the request. The language code depends upon the header information sent by the Web browser. Specify a language code based on the language codes defined in ISO639. Note that you must specify the MultiViews options with the Options directive to enable the content negotiation. When performing multiple specifications of this directive, you cannot specify different language codes for the same extension.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
AddLanguage ja .ja                 Japanese
AddLanguage en .en                 English
AddLanguage fr .fr                 French
AddLanguage de .de                 German
AddLanguage da .da                 Danish
AddLanguage el .el                 Greek
AddLanguage it .it                 Italian
```

(15) AddType *MIME-type extension [extension ...]*

(a) Contents

Specify the AddType directive when you want to associate the extension of the undefined contents and the MIME type in the file specified with the TypesConfig directive. When performing multiple specifications of this directive, you cannot specify different MIME types for the same extension.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
AddType text/html .shtml
```

Associate the MIME type text/html with the extension .shtml.

(16) Alias URL directory-name

(a) Contents

Specify the Alias directive when replacing a specific URL requested from the Web browser with an optional name. However, in the URL you cannot specify the characters from ? (query string) onwards. The directory specified in the URL is replaced with the directory specified in the directory name and displayed on the Web browser.

You cannot specify a URL that is duplicated with the following directive specification values:

- ProxyPass path name
- JkMount URL patterns in the redirector definition file

For example, the following URLs cannot be specified:

```
  Alias /aaa/bbb/ C:/alias/
  ProxyPass /aaa/ http://aaa.example.com/
```

Specify the directory name with an absolute path.

(b) Location where you can code

httpsd.conf,<VirtualHost>

(c) Specification example

```
Alias /icons/ "<Cosminexus-installation-directory>/httpsd/
icons/"
```

Replace /icons/ with <*Cosminexus-installation-directory*>/httpsd/icons/.

(17) AliasMatch regular-expression new-path

(a) Contents

Specify the AliasMatch directive when replacing the URL requested from the Web browser with an optional name. However, in the URL you cannot specify characters from ? (query string) onwards.

When a URL that satisfies the conditions described in the regular expressions is requested from the Web browser, the contents of the specified new path are displayed in the Web browser. When the contents are grouped using bracket () in the regular expressions, you can refer the character strings that match with the expression of group i by using $i in new path. Specify the numeric values from 1 to 9 for i.

You cannot specify a regular expression that is duplicated with the following direcrive specification values:

- ProxyPass path name
- JkMount URL patterns in the redirector definition file

For example, the following regular expressions cannot be specified:

```
  AliasMatch ^/aaa/bbb/(.*) C:/alias/$1
  ProxyPass /aaa/ http://aaa.example.com/
```

Specify the new path with absolute path. When '$' or '&' are included as characters of the new path, add '￥' before these characters. Note that, you need not add '￥' before '$'when you specify $i.

(b) Location where you can code

httpsd.conf,<VirtualHost>

(c) Specification example

```
AliasMatch ^/html/(.*) "C:/htdocs/html/$1"
```

If a request starts with /html/, replace the /html/ portion with C:/htdocs/html/. For example, if you want to access /html/index.html, replace /html/index.html with C:/htdocs/html/index.html.

(18) Allow from {host | all | env = environment-variable} [{Host | env = environment-variable} ...]

(a) Contents

Specify the 'Allow from' directive to restrict the clients that can access the Web server. In the host, you can specify the domain name, IP address, subnet, and net mask of the host to which access is permitted. Specify `all` for permitting access from all the hosts.

The domain name, address, and prefix length of the IPv6 address can also be specified for a host. Do not specify the IPv6 address by enclosing it in square brackets ([ ]). The prefix length is specified in the *IPv6-address/prefix-length* format. Specify the prefix length as a decimal value.

If you specify env=*environment-variable*, you can control server access with the value set for the environment variable. You can restrict access based on the HTTP request header field if you use this directive in combination with the BrowserMatch, BrowserMatchNoCase, SetEnvIf, and the SetEnvIfNoCase directive.

You can use the Order directive to set the evaluation order of the Allow directive (access permission) and Deny directive (access restriction).

Host	Meaning
Domain name	Permit the access from the host specified in the domain name.
IP address	Permit the access from the host specified in the IP address.
Subnet	Permit the access from the host specified in the subnet (First three bytes of the IP address)
Netmask	Permit the access from the host specified in the netmask notation (example: 10.1.0.0/255.255.0.0). When the notation is in 10.1.0.0/16 format, it means same as 10.1.0.0/255.255.0.0.

(b) Location where you can code

<Directory>, .htaccess

(c) Overwrite permission

Limit level

(d) Specification example

(Example 1)

```
SetEnvIf User-Agent Mozilla.* access_ok
<Directory /docroot>
    Order deny,allow
    Deny from all
    Allow from env=access_ok
</Directory>
```

In this case, access is permitted only to the browser requests in which the User-Agent character string contains Mozilla, and access is denied to all other requests.

(Example 2)

When specifying an IPv6 address for a host, the directive is coded as follows:

```
allow from 2001::123:4567:89ab:cdef
```

When specifying a prefix length, all of the following settings are identical:

```
allow from 2001:0:0:89ab::/64
allow from 2001:0:0:89AB::/64
allow from 2001::89ab:0:0:0:0/64
allow from 2001:0000:0000:89ab:0000:0000:0000:0000/64
```

(19) AllowOverride directive [directive ...]

~<<All>>

(a) Contents

The AllowOverride directive specifies whether overwriting of the access information definition with the file specified in AccessFileName directive is permitted. For directives that can be controlled by each directive, see the description of the overwrite permissions for each directive.

Directive	Contents
AuthConfig	AuthGroupFile, AuthName, AuthType, AuthUserFile, and Require directives Permits overwriting of the directives related to the access control of server
FileInfo	AddType, AddEncoding, and AddLanguage directives Permits overwriting of the directives related to file information such as contents management, MIME type, and encryption
Indexes	FancyIndexing, AddIcon, and AddDescription directives Permits overwriting of the directives related to directory index
Limit	Allow from, Deny from, and Order directives Permits overwriting of the access control using the host name or the IP address
Options	Permits the use of Options directory
All	Permits overwriting of all the directives
None	Prohibits overwriting of all the directives

(b) Location where you can code

<Directory>

(20) AuthAuthoritative {On | Off}

(a) Contents

The AuthAuthoritative directive specifies the controlling methods for the user authentication.

On: Performs the users authentication as per the settings of AuthUserFile, AuthGroupFile, and Require directives. When the user is not registered or when the password does not match, an error status 401 is displayed in the Web browser.

Off: Performs the users authentication as per the setting of AuthUserFile, AuthGroupFile, and Require directive. In such cases, if the password is incorrect, an error status 401 is displayed in the Web browser. If the user is not registered, performs the user authentication using the modules (functionality) of other products.

(b) Location where you can code

<Directory>, .htaccess

(c) Overwrite permission

AuthConfig level

(21) AuthGroupFile file-name

(a) Contents

When performing the user authentication in groups, the AuthGroupFile directive specifies a file name that stores the list of the groups to be authenticated. In the file name, you can specify the absolute path or the relative path from the specification value of the ServerRoot directive.

Create the group file in the following format using a text editor:

group-name: *user-name* [*user-name* ...]

User names registered in the password file for user authentication are defined in a group name of your choice. Specify one group per line. You can define multiple groups in the group file. If you specify the same group name on multiple lines, a single group is defined that includes all of the user names that are registered to the same group name.

(b) Location where you can code

<Directory>, .htaccess

(c) Overwrite permission

AuthConfig level

(22) AuthName realm-name

(a) Contents

Specify the realm name (displayed in the user authentication screen of the Web browser) when performing the user authentication. When you specify this directive, you must specify AuthType, Require, and AuthUserFile (or AuthGroupFile) directives. However, when performing the user authentication with the directory service, you need not specify AuthUserFile (or AuthGroupFile) directive.

(b) Location where you can code

<Directory>, .htaccess

(c) Overwrite permission

AuthConfig level

(23) AuthType ***authentication-type-name***

(a) Contents

The AuthType directive specifies the authentication control type when performing the user authentication. You can specify "Basic" as the authentication type. When you specify this directive, you must specify AuthName, Require, and AuthUserFile (or AuthGroupFile) directives. However, when performing the user authentication with the directory service, you need not specify the AuthUserFile (or AuthGroupFile) directive.

Basic: Converts the Base64 code.

(b) Location where you can code

<Directory>, .htaccess

(c) Overwrite permission

AuthConfig level

(24) AuthUserFile ***file-name***

(a) Contents

When you authenticate the user with the user name, specify the name of the file that stores the list of the user names and the passwords to be authenticated.

Specify the file name with an absolute path or the relative path from the value specified for the ServerRoot directive.

(b) Location where you can code

<Directory>, .htaccess

(c) Overwrite permission

AuthConfig level

6.2.3 Directives starting with B, C, and D

*(1) BindAddress {**IP-address** | *}*

~<<*>>

(a) Contents

The BindAddress directive specifies the IP address to which the Web server can be connected, from the allocated IP addresses in the server machine on which the Web server is installed. An IPv6 address cannot be specified for *IP-address*. To enable connection to the Web server from any IPv4 address, specify an asterisk (*). If you specify the Listen directive, the specification of the BindAddress directive is ignored.

(b) Location where you can code

httpsd.conf

(2) BrowserMatch "browser-name" environment-variable[=value] [environment-variable[=value] ...]

(a) Contents

Specify the BrowserMatch directive when you set the environment variables for each Web browser. Default setting value is 1. When ! (exclamation mark) is added before the environment variable, the settings of that environment variable are cancelled. You can specify the case sensitive browser names with the regular expressions.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Specification example

```
BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0
force-response-1.0
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0
BrowserMatch "Microsoft Data Access Internet Publishing
Provider" redirect-carefully
BrowserMatch "^WebDrive" redirect-carefully
BrowserMatch "^WebDAVFS/1.[012]" redirect-carefully
BrowserMatch "^gnome-vfs" redirect-carefully
```

The meaning of environment variables shown in the specification example is as follows:

Environment variable	Contents
nokeepalive	This environment variable disables the KeepAlive connection. You cannot disable the KeepAlive connection when Via header is added in the request.
downgrade-1.0	This environment variable handles the requests of HTTP/1.1 and above as requests of HTTP/1.0.
force-response-1.0	This environment variable always responds with HTTP/1.0 to the requests of HTTP/1.0.
redirect-carefully	When accessing the directory if '/' is not added at the end of the URL, and a method other than the GET method is used, redirect is not requested to the client.

(3) BrowserMatchNoCase "browser-name" environment-variable[=value] [environment-variable[=value] ...]

(a) Contents

Specify the BrowserMatchNoCase directive when you set the environment variables for each Web browser. The default value is 1. When ! (exclamation mark) is added before the environment variable, the settings of that environment variable are cancelled. You can specify the browser name with the regular expressions, and the browser name is not case sensitive.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(4) CacheNegotiatedDocs [{On | Off}]

(a) Contents

The CacheNegotiatedDocs directive specifies whether to enable cache in the client during the request for content negotiation. When you omit the directive arguments, the Web server operates in the same manner as when `On` was specified. If you do not specify the directive, the Web server operates in the same manner as when `Off` was specified. The specification of this directive is invalid for the HTTP/1.1 requests.

On: Contents are cached.

Off: Expires header is added and contents are not cached.

(b) Location where you can code

httpsd.conf

(5) CoreDumpDirectory directory-name U

~<<ServerRoot directive specified value>>

(a) Contents

The CoreDumpDirectory directive specifies the directory for the core dump. You can specify the absolute path or the relative path from the specified value of the ServerRoot directive. Note that the specified directory must have write access to the users and groups that are specified in the User and Group directives. In the Linux version, this directive is applied only when it is specified in the configuration file.

(b) Note

In HP-UX, core dumps for user ID changes are not output. If Cosminexus HTTP Server is started by a superuser, the user is changed to the user specified by the User directive. Because of this, no core dump is output even if an error that normally results in a core dump occurs.

(c) Location where you can code

httpsd.conf

(6) CustomLog {file-name | pipe} {"format" | label-name} [env=[!]environment-variable]

(a) Contents

Specify the CustomLog directive when the log in any format is to be output to the file. The format is similar to the one that is specified in the LogFormat directives.

When specifying multiple specifications of this directive, you cannot specify the same file name multiple times.

file-name: Specifies the log file name. In the file name, you can specify the absolute path or the relative path from the specified value of the ServerRoot directive.

pipe: Specifies the program that receives log information from the standard input in the "| Program name" format. The Web server converts the linefeed codes contained in the log information to CRLF and passes it on.

(Notes for Windows version)

Each program specified in the pipe is generated as a separate process that receives the log information for the control process and Web server process. This process is called the *Pipe process*. Note the following points when you start the Web server as a service:

- Unable to obtain the log information of the control process

  When you start the Web server as a service, the standard input for receiving the log information from the control process is associated with the NUL device, and hence the pipe process for control cannot receive the log information from the control process. The log information from the control process refers to the error log information when the Web server starts and stops, and you cannot collect this information. The information of the error log and the access log after starting the Web server is the log information from the Web server process, and hence the information is received in the pipe process for the Web server process.

- Points to be remembered during program creation

  The pipe process used for the control process is the process for reading the data from the NUL device, where the pipe process has a small read() buffer and the input data is in the waiting state. Set the read() buffer with a bigger value so that the pipe process is not in the input data waiting state.

- When an argument is specified in the program

  When the programs and the arguments contain a blank, enclose them with a ¥".

  When the programs and arguments are enclosed with ¥", enclose the entire statement with a ¥".

(Example)

```
CustomLog "|￥"￥"<Cosminexus-installation-directory>/
httpsd/sbin/rotatelogs.exe￥"
￥"<Cosminexus-installation-directory>/httpsd/logs/
access￥" 86400 -diff 540￥"" common
```

- When an argument is not specified correctly in the program

  If an argument is not specified correctly in the program, the program fails to start but the Web server starts properly. In this case, logs are not output. When you specify a program, make sure that the log files are created and the files are divided as you intended.

"*Format*": Specifies the log format. the following tables describe the formats that you can specify.

label-name: Specifies the label name that is specified in the LogFormat directive.

env=*environment-variable*: Collects the log when the specified environment variable is set.

env=!*environment-variable*: Collects the log when the specified environment variable is not set.

Table 6-3: The format list

Format	Meaning
%A#1	IP address of the Web server.
%a#1	IP address of the client.
%B	Number of bytes sent (excluding the data added by the HTTP header and chunked encoding).
%b	Number of bytes sent (excluding the data added by the HTTP header and chunked encoding). However in the case of 0, - (hyphen).
%{*cookie_name*}C	Value of the cookie name *cookie_name* included in the Cookie header value.When multiple instances of cookie_name are found in the Cookie header value, all values are output.
%D	Display the request processing time in microseconds.
%{*env_name*}e	Value of the environment variables specified in *env_name*.
%f	Directory or the file name requested by the client.
%H	Request protocol (such as HTTP/1.0).
%h	Host name of the client.

Format	Meaning
%I	Total number of bytes received including the request and the header.
%{*header_name*}i	Value of the request header specified in *header_name*.
%l	Identification information of the client (when IdentityCheck directive is On, and identd is running on the client).
%m	Request method (such as GET, POST).
%{*note_name*}n	Value of the note of Web server module specified in the *note_name*.
%O	Total number of bytes sent including header.
%{*header_name*}o	Value of the response header specified in *header_name*.
%P	Process ID that processes the request of HTTP communication.
%{*hws_thread_id*}P	Thread ID that processes the request of HTTP communication. Valid for Windows version.
%p	Port number.
%q	Query character string.
%r	First line of the HTTP communication request.
%s	Status (for the logs that are internally redirected, original status is shown).
%T	Time taken for request processing (seconds). If `On` is specified in the HWSLogTimeVerbose directive, then time is displayed up to milliseconds.
%t	Time when request processing starts. If `On` is specified in the HWSLogTimeVerbose directive, then time is displayed up to milliseconds.
%{*format*}t	Time when request processing starts. The format defined by strftime() is specified in *format*.
%U	The URL.
%u	User name of client (when the user authentication is performed).
%V	The ServerName directive specification value, server name, or IP address as per the specifications of the UseCanonicalName directive.
%v	Server name.
%X	Connection status when the response ends. +: Connected even after sending the response. -: Disconnected after sending the response. X: Disconnected before the response ends.
%>s	End status.

#

{ } displayed in the format does not denote the selection. Bold characters in the { } denote the variable name that collects the log and small characters describes the character string without changing it.

#1

If %A or %a is specified in the format, IPv6 addresses can also be output.

#2

If %h or %V is specified in the format, IPv6 addresses or host names corresponding to IPv6 addresses can also be output.

Table 6-4: The log format list related to SSL

Format	Meaning
%{version}c	SSL version
%{cipher}c	Encryption type used in the current communication
%{clientcert}c	Distinguished Name of the subject of the SSL client certificate.

You can describe the status code after the % in the format.

(Example) For error status codes 400 and 501, collect the log of User-Agent request header value.

```
%400,501 {User-Agent}i
```

(Example) For error status codes other than 200, 304, and 302, collect the log of the Referer request header value.

```
%!200,304,302 {Referer}i
```

Specify env= when you divide the log collection as per the settings of the specified environment variables.

(Example) Collect the log for the gif image access in gif.log, and collect the log for other accesses in nongif.log.

```
SetEnvIf Request-URI \.gif$ gif-image
CustomLog logs/gif.log common env=gif-image
CustomLog logs/nongif.log common env=!gif-image
```

(b) Locations where you can code

httpsd.conf,<VirtualHost>

(c) Specification example

```
CustomLog logs/access.log common
CustomLog logs/ssl.log "%t %{version}c %{cipher}c
%{clientcert}c"
```

(7) DefaultIcon URL

(a) Contents

The DefaultIcon directive specifies the icons displayed in the directory index. Specify the URL of the icon to be displayed when the specified icons do not correspond to any of the AddIcon, AddIconByType, and the AddIconByEncoding directives.Note that if you specify the IPv6 address without omitting http://*IP-address* of the URL, enclose the IPv6 address in square brackets ([ ]).

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
DefaultIcon /icons/unknown.gif
```

(8) DefaultLanguage language-code

(a) Contents

The DefaultLanguage directive specifies the default language used in the document. The specified language code is set in the Content-Language response header. The specified language code becomes the default value for the settings of the AddLanguage directive. If the default value is not set, the response header of the Content-Language is not sent.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <directory>, .htaccess

(c) Overwrite permission

FileInfo level

(9) DefaultType MIME-type

~<<text/plain>>

(a) Contents

The DefaultType directive specifies the MIME type name used for the contents that do not correspond to any of the MIME types defined in the file specified with the TypesConfig directive.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
DefaultType text/plain
```

(10) Deny from {host | all | env=environment-variable} [{Host | env=environment-variable} ...]

(a) Contents

Specify the 'Deny from' directive when restricting the clients that can access the Web server. In the host, you can specify the domain name, IP address, subnet, and the netmask of the host that permits the access. Specify `all` when prohibiting the access from all hosts.

The domain name, address, and prefix length of the IPv6 address can also be specified for a host. Do not specify the IPv6 address by enclosing it in square brackets ([ ]). The prefix length is specified in the *IPv6-address*/*prefix-length* format. Specify the prefix length as a decimal value.

If you specify env=*environment-variable*, you can control server access with the value set for the environment variable. You can restrict access based on the HTTP request header field if you use this directive in combination with the BrowserMatch, BrowserMatchNoCase, SetEnvIf, and the SetEnvIfNoCase directive.

You can use the Order directive to set the evaluation order for the Allow directive (access permission) and the Deny directive (access restriction).

Host	Meaning
Domain name	Prohibit access from the host displayed in the domain name
IP address	Prohibit access from the host displayed in the IP address
Subnet	Prohibit access from the host specified in the subnet (First three bytes of the IP address).
Netmask	Prohibit access from the host specified in the netmask (Example: 10.1.0.0/255.255.0.0). If the notation is in 10.1.0.0/16 format, it means same as 10.1.0.0/255.255.0.0.

(b) Location where you can code

<Directory>, and .htaccess

(c) Overwrite permission

Limit level

(11) DirectoryIndex file-name [file-name ...]

~<<index.html>>

(a) Contents

When the requests from the Web browser do not specify any specific contents, the DirectoryIndex directive specifies the default file name of the contents to be sent to the client. If you specify multiple file names, file name specified first will be sent first.

When the specified file is not present in the requested directory, the Web browser display changes according to the Options directive.

- When Indexes are enabled

  The Web browser displays the index of the directory created in the Web server.
- When Indexes are disabled

  The Web browser responds with the status code 403 Forbidden.

(b) Location where you can code

httpsd.conf,<VirtualHost>,<Directory>,.htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
DirectoryIndex index.html
```

If a file name is not specified in the request, display index.html file if present in the directory).

(12) DocumentRoot directory-name

~<</opt/hitachi/httpsd/htdocs>> (UNIX version)

~<<ServerRoot directive default value/htdocs>> (Windows version)

(a) Contents

The DocumentRoot directive specifies the document root directory that stores the contents, with an absolute path. Do not add / (slash) at the end of the directory name.

Specify the directory name with an absolute path.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
DocumentRoot "<Cosminexus-installation-directory>/httpsd/
htdocs"
```

6.2.4 Directives that start with E, F, G, H, and I

(1) ErrorDocument error-status-number {Text **|** *local-URL* **|** *full-URL}*

(a) Contents

Specify the ErrorDocument directive to customize the error message displayed on the Web browser when an error occurs.

Text: Specifies the character string enclosed in ".

local-URL: Specifies the contents of the local site by adding a / at the beginning.

full-URL: Specifies the contents of another site by using a URL that starts with http:// or https://.

The error status numbers that can be specified in this directive and the possibility to specify text, the local URL, and the full URL is described below:

Error status number (meaning)	Text	Local URL	Full URL
400 (Bad Request)	Y	N	N
401 (Authorization Required)	Y	Y	N
403 (Forbidden)	Y	Y	Y
404 (Not Found)	Y	Y	Y
405 (Method Not Allowed)	Y	Y	Y
406 (Not Acceptable)	Y	Y	Y
408 (Request Time-out)	N	N	N
410 (Gone)	Y	Y	Y
411 (Length Required)	Y	N	N
412 (Precondition Failed)	Y	Y	Y
413 (Request Entity Too Large)	Y	Y	Y
414 (Request-URI Too Large)	Y	N	N
416 (Requested Range Not Satisfiable)	Y	Y	Y
417 (Expectation Failed)	Y	N	N
500 (Internal Server Error)	Y	Y	Y
501 (Method Not Implemented)	Y	Y	Y
502 (Bad Gateway)	Y	Y	N

Error status number (meaning)	Text	Local URL	Full URL
503 (Service Temporarily Unavailable)	Y#	Y#	Y#
506 (Variant Also Negotiates)	Y	Y	Y

Legend:

Y: can be specified.

N: cannot be specified.

#

Use the QOSResponse directive or the QOSRedirect directive when customizing the message returned by the flow restriction functionality.

When you specify this directive, consider the following points:

- When performing multiple specifications of this directive, you cannot specify different specification for the same error number.
- You cannot customize the messages for error status set in the CGI programs.
- You cannot customize the message if error occurs in the specification destinations of the local URL and the full URL.
- Error occurs when content negotiation takes place in the specification destination of the local URL and you cannot customize the message.
- You may not be able to customize the message (depending on the mounting method of the module) even with the error status set in the module that is dynamically connected by the LoadModule directive.
- When specifying a full URL, the status code 302 Found and a response with the new path set in the Location header are returned. Normally, a Web browser that receives status code 302 automatically redirects the request to the address specified in the Location header.
- When specifying a full URL, you can also specify an IPv6 address or the host name corresponding to an IPv6 address. When specifying the IPv6 address, enclose the IPv6 address in square brackets ([ ]).

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
ErrorDocument 500 "Server Error."
ErrorDocument 404 /missing.html
ErrorDocument 403 http://some.other_server.com/
subscription_info.html
ErrorDocument 404 http://[2001::123:4567:89ab:cdef]/
missing.html
```

(2) ErrorLog {file-name | pipe}

~<<logs/error_log>> (UNIX version)

~<<logs/error.log>> (Windows version)

(a) Contents

The ErrorLog directive specifies the name of the file in which error log is output. You can select the log contents to be output by the LogLevel directive.

In the file name you can specify absolute path, or relative path from the specified value of the ServerRoot directive.

file-name: Specifies the name of the file that stores the error log. You can specify the file name using the relative path from the specified value of the ServerRoot directive.

pipe: Specifies the program that receives error log information from the standard input in the "|Program name" format. For the notes on Windows version, see *CustomLog directive*.

(b) Location where you can code

httpsd.conf, and <VirtualHost>

(c) Specification example

```
ErrorLog logs/error.log
```

(3) ExtendedStatus {On | Off}

(a) Contents

The ExtendedStatus directive specifies whether to display the extended status information of each request in the format for displaying the status by the server-status handler.

On: Displays extended status information. This information is displayed even if the client IP address is an IPv6 address. A maximum of 31 bytes can be displayed.

Off: Does not display the extended status information.

(b) Location where you can code

httpsd.conf

(4) ExpiresActive {On | Off}

(a) Contents

The ExpiresActive directive specifies whether to add the Expires header and Cache-Control header to the response.

On: Adds the Expires header and Cache-Control header.

Off: Does not add the Expires header and Cache-Control header.

(b) Note

- The mod_expires module must be embedded to use the expiry date settings functionality. For details on the expiry date settings functionality, see *4.11 Functionality to set expiry date*.
- In the case the ExpiresDefault directive or the ExpiresByType directive is not specified, the Expires header and Cache-Control header are not added to the response even if `On` is specified in the ExpiresActive directive.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(d) Overwrite permission

Indexes level

(5) ExpiresByType MIME-type {A | M} time

~((0-2147483647))(Unit: Seconds)

(a) Contents

The ExpiresByType directive specifies the expiry date for the specified MIME type document when you add Expires header and Cache-Control header to the response. This directive is enabled when the ExpiresActive directive is set to `On`. The default expiry date set in the ExpiresDefault directive is overwritten by these settings for each MIME type.

Specify the standard time by A or M, and specify the time from the standard time to the expiry date in seconds. Do not enter a space between A or M and the time.

A: The time when client accesses is interpreted as the standard time.

M: The time when the file was last modified is interpreted as the standard time.

(b) Note

- The mod_expires module must be embedded to use the expiry date settings functionality. For details on the expiry date settings functionality, see *4.11 Functionality to set expiry date*.
- Set the expiry date such that it is not after January 19, 2038, 03:14:07 of the

Greenwich Mean Time (GMT).

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(d) Overwrite permission

Indexes level

(e) Specification example

```
ExpiresByType text/html A604800
```

(6) ExpiresDefault {A | M} time

~((0 - 2147483647)) (Unit: Seconds)

(a) Contents

The ExpiresDefault directive specifies the default expiry date when you add the Expires header and Cache-Control header to the response. This directive is enabled when the ExpiresActive directive is set to `On`. The ExpiresByType directive overwrites these settings for each MIME type.

Specify the standard time by A or M, and specify the time from the standard time to the expiry date in seconds. Do not enter a blank between A or M and the time.

A: The time when the client accesses is interpreted as the standard time.

M: The time when the file was last modified is interpreted as the standard time.

(b) Note

- The mod_expires module must be embedded to use the expiry date settings functionality. For details on the expiry date settings functionality, see *4.11 Functionality to set expiry date*.
- Set the expiry date such that it is not after January 19, 2038, 03:14: 07 of the Greenwich Mean Time (GMT).

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(d) Overwrite permission

Indexes level

(e) Specification example

```
ExpiresDefault A604800
```

(7) FancyIndexing {On | Off}

(a) Contents

The FancyIndexing directive specifies whether to perform the format display (fancy index) when you display the directory index.

On: Performs the format display.

Off: Does not perform the format display.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
FancyIndexing On
```

Use the format display functionality.

(8) FileETag [{ + | - }]option [[{ + | - }]option ...]

~<<All>>

(a) Contents

The FileETag directive specifies the file attribute value used for creating Etag response header field. When this directive is not specified, the unique ID assigned to the file, last updated time, and bytes count are set in the Etag response header field.

When you do not specify + **-** in the option, the attribute value specified in the option is used.

When you specify + **-** in the option, you can change the attribute value set by the FileETag directive.

+: The attribute value specified in the option is added to the set attribute value.

-: The attribute value specified in the option is deleted from the set attribute value.

The following table lists the options that you can specify:

Option	Meaning
Inode	The unique ID assigned to the file is included.
Mtime	Last updated time of the file is included.
Size	Bytes count of the file is included.
All	All the Inode, Mtime, and the Size options are enabled.

Option	Meaning
None	Etag header is not added.

(b) Notes

- When you enable the Inode option of the FileETag directive, different IDs may be included in the Etag header every time when same contents are requested in the Web server environment where load balancing is performed. Consequently, it may be inconvenient for caching in the browser and the proxy since Etag header is different although the contents are same. You can avoid such a situation by disabling the Inode option with the FileETag directive.
- When performing multiple specifications of this directive without using + - options only the directive specified in the end is enabled.
- When only the attribute value with the - option is specified, the operation is same as the case where the `All` option is specified.
- For the `All` option and the `None` option, you cannot specify + -.
- If the -Inode, -Mtime, and -Size options are specified together, the status is the same as when this directive is not specified. The inode number, the latest update time, and the number of bytes in the file are set in the ETag response header field.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(d) Overwrite permission

FileInfo level

(e) Specification example

(Example 1)

```
FileETag Inode Mtime Size
FileETag -Inode

In this specification, the last updated time of the file
and the byte count are used as the attribute value.
```

(Example 2)

```
FileETag Inode Mtime
FileETag Size

In this specification, byte count of the file is used
as the attribute value.
```

(Example 3)

```
FileETag All
FileETag -Inode -Mtime -Size
```

```
In this specification, unique ID of the file, last
updated time, and byte count are used as the attribute
value.
```

(9) ForceType MIME-type

(a) Contents

Define the ForceType directive in the <Directory> block or in the access control file. This directive specifies the MIME types to be used for all the contents under the specific directory. If you specify `none`, the previous ForceType directive specifications are disabled.

(b) Location where you can code

<Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(10) Group group-name U

~<<#-1>>

(a) Contents

This directive specifies the group name used when server processes are running.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
Group nogroup                    Define the group name nogroup
```

(11) Header {{set | append | add} header header-value [env=[!]environment-variable] | unset header}

(a) Contents

Specify the 'Header' directive for customizing the response header when the Web server responds with the status code 200. When using this directive as a reverse proxy, the response header is customized regardless of the status code value returned by the backend Web server.

set: Sets the header. If the header is found, rewrite it with the specified header value.

append: Adds the header value to the existing header. A comma delimits the existing header values. Sets the header if it does not exist.

add: Sets a header in another line even if the header exists. Use this directive when setting the same header in multiple lines.

unset: If the specified header exists, deletes it.

env=*environment-variable*: When the specified environment variable is set, executes the contents specified in the Header directive.

env=!*environment-variable*: When the specified environment variable is not set, executes the contents specified in the Header directive.

If the header value contains spaces, then you must enclose the value in "(double quotation mark). For the header value you can specify character strings containing only the characters, character string including the format identifiers, or the character string containing both the characters and the format identifiers. The format identifiers are as follows:

Format	Meaning
%t	Display the time when the request was received, by the time elapsed from January 01, 1970, 00:00:00 (GMT: Greenwich Mean Time. The unit is Microseconds. "t=" is added in the beginning.
%D	Display the time taken for request processing. The unit is microseconds. "D=" is added in the beginning.
%{*env_name*}e	Value of the environment variable env_name.

(b) Note

The mod_headers module must be embedded to use the header customization functionality. For details on the header customization functionality, see *4.10 Header customization functionality*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(d) Overwrite permission

FileInfo level

(e) Specification example

```
Header set Cache-Control no-cache
```

(12) HeaderName file-name

(a) Contents

The HeaderName directive specifies the file name (without path information) of the file that describes the comments added to the header when displaying the directory index. You can describe the file name in the HTML or the plain text format. However, MIME type must be correctly defined in the file specified with the AddType directive or the TypesConfig directive. When you create comments in the plain text, the <PRE> tag is added to the HTML of the directory index display.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
HeaderName HEADER.html
```

Contents of HEADER.html under each directory are added to the header.

(13) HostnameLookups {On | Off | double}

(a) Contents

The HostnameLookups directive specifies whether to reverse the lookup of the host name to convert the IP address of the REMOTE_HOST environment variable of CGI and the client IP address to be output to the log file, into the host name. If you use reverse, response is delayed.

On: Converts the IP address into the host name.

Off: Does not convert the IP address into the host name.

double: Converts the IP address into the host name. After conversion, reconvert and confirm that the IP address is correct.

This directive supports IPv6 addresses.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>

(c) Specification example

```
HostnameLookups Off
```

Do not convert the IP address in the host name.

(14) HWSContentCacheMaxFileSize size W

~((1 - 2097093))<<256>> (Unit: KB)

(a) Contents

The HWSContentCacheMaxFileSize directive specifies the upper limit of the file size that you can cache in KB. In the HWSContentCacheMaxFileSize directive, if you specify a value greater than the value of the HWSContentCacheSize directive, the value of the HWSContentCacheSize directive is set.

(b) Note

The mod_hws_cache module must be embedded to use the static contents cache functionality. For details on the static contents cache functionality, see *4.12 Static contents cache functionality*.

(c) Location where you can code

httpsd.conf

(d) Specification example

```
HWSContentCacheMaxFileSize 32
```

(15) HWSContentCacheSize size W

~((1 - 2097093))<<8192>> (unit: KB)

(a) Contents

The HWSContentCacheSize directive specifies the upper limit of the memory size of the data to be cached in server process, in KB.

(b) Note

The mod_hws_cache module must be embedded to use the static contents cache functionality. For details on the static contents cache functionality, see *4.12 Static contents cache functionality*.

(c) Location where you can code

httpsd.conf

(d) Specification example

```
HWSContentCacheSize 1024
```

(16) HWSErrorDocumentMETACharset {On | Off | character-set} W

(a) Contents

This directive sets the character set for messages (hereafter called the *error document*) that are displayed on the Web browser when an error occurs. In the error document, the character set is set as a value of `charset=` by the META tags. For the error documents customized with the ErrorDocument directive, character set is not set as per the META tags in this directive.

On: Sets the character set ISO-8859-1.

Off: Does not set the character set.

character-set: Sets the specified character set.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
HWSErrorDocumentMETACharset ISO-2022-JP
```

(17) HWSErrorLogClientAddr X-Forwarded-For

(a) Contents

Specify The HWSErrorLogClientAddr directive to change the message text to be output to error log from"[client *client-address*]" to "[X-Forwarded-For *header-value*]".On the back-end server, change the the message text to be output to the error log from "[client *client-address*]" to "[X-Forwarded-For *header-value*]".

When the back-end server receives a request through a load balancer or a proxy server, "[client *client-address*]" to be output to the error log could becomes the IP address of the load balancer or proxy server, instead of the actual client IP address from which the request was sent. However, as because some load balancers and proxy servers can add the original client IP address to the X-Forwarded-For header, change the header settings to the client IP address by changing the output content to the value of the X-Forwarded-For header.

X-Forwarded-For: Change "[client *client-address*]" to be output to error log to "[X-Forwarded-For *header-value*]".

(b) Notes

Some types of messages cannot be changed. For example, when an error occurs before the server receives the X-Forwarded-For header. The messages you cannot change are as follows.

- Messages for basic functionality

No.	Message
1	[client *client-address*] *detailed-information*: core_output_filter: Error reading from bucket.
2	[client *client-address*] request failed: URI too long (longer than *upper-limit*)
3	[client *client-address*] request failed: erroneous characters after protocol string: *request-line*
4	[client *client-address*] client sent invalid HTTP/0.9 request: HEAD *request-URI-value*
5	[client *client-address*] {child process *process-ID*|server thread *thread-ID*}:forcing termination of request "*request-line*"
6	[client *client-address*] *detailed-information*: core_output_filter: writing data to the network
7	[client *client-address*] request failed: error reading the headers#

#

This message is to show that the reading of the header is in progresss. You can change this message only when the X-Forwarded-For header is already read.

- SSL messages

No.	Message
1	[client *client-address*] [port *client-port-number*] allocate error
2	[client *client-address*] [port *client-port-number*] data set error
3	[client *client-address*] [port *client-port-number*] No client certificate
4	[client *client-address*] *detailed-information*: SSL handshake interrupted by system: client port *port-number*
5	[client *client-address*] *detailed-information*: SSL handshake interrupted by system: client port *port-number* (*SSL-handshake-process-time*)(*error-number-value*)(*server-process-ID*): *SSL-handshake-processing-status*
6	[client *client-address*] SSL library error *error-number* in handshake
7	[client *client-address*] [port *client-port-number*] SSL library error *error-number* in handshake(*SSL-handshake-process-time*)(*error-number-value*)(*server-process-ID*): *SSL-handshake-processing-status*
8	[client *client-address*] [port *client-port-number*] verify error:num=*value*:*error-message*
9	[client *client-address*] [port *client-port-number*] Verify depth exceeded
10	[client *client-address*] [port *client-port-number*] verify error

(c) Location where you can code

httpsd.conf

(18) HWSGracefulStopLog {On | Off}

(a) Contents

The HWSGracefulStopLog directive specifies whether the request information that is forcefully terminated after the waiting time of forced termination elapses, is to be output in the error log file when you execute planned termination.

On: Outputs the forcefully terminated request information to the error log file.

Off: Does not output the forcefully terminated request information to the error log file.

(b) Location where you can code

httpsd.conf

(c) Specification example
```
HWSGracefulStopLog On
```

(19) HWSGracefulStopTimeout forced-termination-time

~((0 - 3600))<<300>> (Unit: Seconds)

(a) Contents

The HWSGracefulStopTimeout directive specifies the forced termination waiting time until the request being executed is stopped at once during the planned termination. The upper limit of the forced termination waiting time is not set if `0` is specified.

(b) Location where you can code

httpsd.conf

(c) Specification example
```
HWSGracefulStopTimeout 600
```

(20) HWSImapMenuCharset character-set

(ISO-8859-1)

(a) Contents

The HWSImapMenuCharset directive specifies the character set displayed in menus in the following cases:

- When `map` is specified for the value of the image map file
- When the mouse is used to point to coordinates (0,0) on an image map
- When an image map file is requested without any coordinates specified

The character set is set as the value of `charset=` in the Content-Type header of the response.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example
```
HWSImapMenuCharset SHIFT_JIS
```

(21) HWSKeepStartServers {On | Off} U

(a) Contents

The HWSKeepStartServers directive specifies whether to maintain only the number of running server processes that are specified in the StartServers directive.

On: Only the number of running server processes that are specified in the StartServers directive are maintained. If the number of server processes is less than the value specified in the StartServers directive, new processes are generated. This functionality is valid when the value specified in each directive regarding the number of processes, is related as follows:

MinSpareServers < StartServers ≤ MaxClients

and

MinSpareServers < MaxSpareServers ≤ MaxClients

When the setting value of the StartServers directive is less than the setting value of the MinSpareServers directive, the number of server processes is maintained by the value of the MinSpareServers directive.**Off**: The running server processes, equivalent to the number specified in the StartServers directive are not maintained.

For details on other directives regarding the number of processes, see *4.1 Relationship between processes and directives of Cosminexus HTTP Server*.

(b) Location where you can code

httpsd.conf

(22) HWSLogSSLVerbose {On | <u>Off</u>}

(a) Contents

The HWSLogSSLVervose directive specifies whether to output detailed information for the info-level and error-level failures among the errors that are output to the log during the SSL handshake process between clients and servers. When SSL is enabled, we recommend that you set this directive to `On`.

On: Displays detailed information

Off: Does not display detailed information

(b) Location where you can code

httpsd.conf

(23) HWSLogTimeVerbose {On | <u>Off</u>}

(a) Contents

The HWSLogTimeVerbose directive specifies whether to display error log# times, request log times, access times in the access log, the time taken for request processing (%), and request process start times (%t), accurate to the millisecond.

#: This directive targets the error log specified by using the ErrorLog directive. The CGI script error log specified by using the ScriptLog directive is not targeted.

On: Displays the hours and time in Milliseconds.

Off: Displays the hours and time accurate to the second

(b) Location where you can code

httpsd.conf

*(24) HWSMaxQueueSize **request-queue-size*** W

~((0 - 2147483647))<<8192>>

(a) Contents

The HWSMaxQueueSize directive specifies the maximum number of waiting requests for requests from clients. There is no limit on the number of requests when `0` is specified. The requests from the client that exceed the request queue size specified in this directive are disconnected in the server side.

(b) Location where you can code

httpsd.conf

*(25) HWSNotModifiedResponseHeaders **header-name [header-name ...]***

(a) Contents

The HWSNotModifiedResponseHeaders directive specifies the response header added when the status code 304 Not Modified is sent to the client.

Note that the headers below are added to responses even if they are not specified in this directive. However, the headers are not always added, but are added only when settings for the headers have been performed, such as on external modules or servers.

- Date
- Server
- Connection
- Keep-Alive
- ETag
- Content-Location
- Expires
- Cache-Control
- Vary
- Warning
- WWW-Authenticate

- Proxy-Authenticate

(b) Location where you can code

httpsd.conf

(c) Specification example

```
HWSNotModifiedResponseHeaders Set-Cookie Set-Cookie2
```

(26) HWSProxyPassReverseCookie path-name

(a) Contents

When using a reverse proxy, the HWSProxyPassReverseCookie directive enables the reverse proxy to convert a Set-Cookie header received from a backend server. This process is required to send the cookie as the return for a request sent to the backend server via the reverse proxy after the Web browser receives the Set-Cookie header.

path-name: Specify the same name as for the ProxyPass directive.

(b) Note

The mod_proxy and mod_proxy_http modules must be embedded to use a reverse proxy. For details on reverse proxies, see the *4.7 Setting the reverse proxy*.

(c) Location where you can code

httpsd.conf, <VirtualHost>

(27) HWSRequestLog {file-name **|** *pipe}*

(a) Contents

The HWSRequestLog directive specifies the name of the file to which a request log is output. The request log is a generic name that refers to module trace information, request trace information, and I/O filter trace information. The type of the output request log can be selected by using the HWSRequestLogType directive.

file-name: Specifies the name of the file to which the request log is output. For the file name, you can specify either an absolute path, or a relative path from the specification value of the ServerRoot directive.

pipe: Specifies the program that receives log information from standard input in the format **|** *program-name*. For notes on the Windows version, see the CustomLog directive.

(b) Notes

- If you omit this directive, the module trace information is output to the file specified by using the ErrorLog directive. Specify the collection level of the module trace information by using the LogLevel directive. For details on module trace information, see *4.2.6 Collecting the module trace*.

- The file specified by using the ErrorLog directive cannot be specified as the output destination for request trace information and I/O filter trace information.

(c) Location where you can code

httpsd.conf

(28) HWSRequestLogType trace-type [trace-type...]

(module-info request)

(a) Contents

The HWSRequestLogType directive specifies the type of trace information to be output to the request log that is set by using the HWSRequestLog directive. The following table shows the types of trace information.

Trace type	Description
module-debug	Outputs trace information for internal modules and the trace information corresponding to module-info. Because specifying this trace type results in the output of a large amount of data, do not specify this trace type for a purpose other than debugging.
module-info	Outputs the module trace information collected when external modules and CGI programs are executed.
request	Outputs trace information when a request process starts and is completed. For a KeepAlive connection, trace information is also output when the next request line is received. This trace is called a *request trace*.
filter	Outputs I/O filter trace information that indicates the execution trigger of the input and output filter function for a module. Because specifying this trace type results in the output of a large amount of data, do not specify this trace type for a purpose other than debugging.
none	No request logs are collected.

(b) Note

When none is included in the specified trace types, no request logs are collected.

(c) Location where you can code

httpsd.conf

(29) HWSSetEnvIfIPv6 request-value IPv6- address environment-variable [= value] [environment-variable [= value] ...]

(a) Contents

The HWSSetEnvIfIPV6 directive defines environment variables based on the IPv6 address of the client or the server. Set the specified environment variable when the request value meets the conditions of the IPv6 address. By default, the value is set to 1. When an exclamation mark (!) is added before an environment variable, that environment variable setting is canceled.

The following request values can be specified.

Request value	Description
Remote_Addr	The IPv6 address of the client
Server_Addr	The IPv6 address of the server that received the request

Specify the IPv6 address without enclosing it in square brackets ([ ]). Note that the prefix length can be specified in decimal format after the IPv6 address. The prefix length is specified in the *IPv6-address*/*prefix-length* format.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
HWSSetEnvIfIPv6 Remote_Addr 2001:0:0:1230::/64 IPV6_CLIENT
```

The IPv6 address of the client starts with 2001:0:0:1230, specify the IPV6_CLIENT environment variable.

(30) HWSStackTrace {On | Off} U

(a) Contents

The HWSStackTrace directive specifies whether to output the contents of the stack trace information. This directive is enabled only when using the HP-UX version.

On: Outputs the contents of the stack trace information to the error log file.

Off: Does not output the contents of the stack trace information to the error log file.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
HWSStackTrace On
```

(31) HWSSuppressModuleTrace module-file-name [all | hook | handler]

(a) Contents

The HWSSuppressModuleTrace directive specifies the file name of a module for which trace information is to be suppressed and the type of functions to be suppressed.

all: Suppresses all module trace information output by the specified module.

hook: Of the trace information output by the specified module, suppresses trace information other than that for the handler function. For details on types of functions, see Table 4-4 of *4.2.6 Collecting the module trace*.

handler: Of the trace information output by the specified module, suppresses only the trace information for the handler function. For details on types of functions, see Table 4-4 of *4.2.6 Collecting the module trace*.

The module file name to be output to either the error log or request log is specified for *module-file-name*. To suppress the module trace information shown in the following example, mod_example.c is specified for *module-file-name*:

(Example)

```
[Mon Dec 18 14:57:14 2006] [info] hws : module -->
(mod_example.c[12])(1896)
[Mon Dec 18 14:57:14 2006] [info] hws : module <--
(mod_example.c[12])(1896)(-1)
```

The following table shows the names of the standard external modules provided by Cosminexus HTTP Server and the corresponding module file names:

Table 6-5: TableNames of standard external modules provided by Cosminexus HTTP Server and corresponding module file names

Module name	Module file name
mod_expires.so	mod_expires.c
mod_headers.so	mod_headers.c
mod_hws_cache.so	mod_hws_cache.c
mod_hws_ldap.so	mod_hws_ldap.c
mod_hws_qos.so	mod_hws_qos.c
mod_proxy.so	mod_proxy.c
mod_proxy_http.so	Module trace information is not output.

When using external modules other than the standard modules provided by Cosminexus HTTP Server, module trace information might be output. In addition, if debug is set by using the LogLevel directive, or if module-debug is set by using the HWSRequestLogType directive, trace information for internal modules is output.

Note that you can specify this directive multiple times. If you specify the directives by using the same module name, only the last directive specified is valid.

(b) Note

Module trace information cannot be suppressed during execution of a CGI program.

(c) Location where you can code

httpsd.conf

(d) Specification example

(Example 1)

```
HWSSuppressModuleTrace mod_proxy.c all
```

In this specification, module trace information for all functions in the proxy module is suppressed.

(Example 2)

```
HWSSuppressModuleTrace mod_proxy.c hook
```

In this specification, module trace information for all functions other than the handler function in the proxy module is suppressed.

(32) HWSTraceIdFile file-name

~<<logs/hws.trcid>>

(a) Contents

The HWSTraceIdFile directive specifies the file name that stores the shared memory ID for trace collection. In the file name you can specify the absolute path, or the relative path from the specified value of the ServerRoot directive.

Multiple Web servers cannot share this file. In the case of starting multiple Web servers by specifying the same ServerRoot directives, you need to specify different file names in this directive.

(b) Location where you can code

httpsd.conf

(33) HWSTraceLogFile file-name

~<<logs/hws.trclog>>

(a) Contents

The HWSTraceLogFile directive specifies the file name that outputs the trace collected in the shared memory when the server process ends abnormally. In the file name you can specify the absolute path, or the relative path from the specified value of the ServerRoot directive.

The trace is output by wrapping around to multiple files.

For the UNIX version, up to 5 files are output. Output file names are *specified-file-name*`.`*nn*, where *nn* is a number from 01 to 05. When Cosminexus HTTP Server starts, *specified-file-name*.01 is the current output file name. When trace information is output to the current output file with file name *specified-file-name*`.`*nn*, the next current file name becomes *specified-file-name*`.`*nn* + 1. If the *nn* portion of *specified-file-name*`.`*nn* is 05, the next current file name becomes *specified-file-name*`.01`.

For the Windows version, up to 2 files are output. Output file names are *specified-file-name*.01 or *specified-file-name*.02. When Cosminexus HTTP Server starts, *specified-file-name*.01 is the current output file name. If trace information is output to the current output file with file name *specified-file-name*.01, the next current file name becomes *specified-file-name*.02. If trace information is output to the current output file whose name is *specified-file-name*.02, the next current file name becomes *specified-file-name*.01.

(b) Location where you can code

httpsd.conf

(34) IdentityCheck {On | <u>Off</u>} U

(a) Contents

The IdentityCheck directive specifies whether to use the `identd` daemon of the client host to check the identity of the client. For details on ident, see *RFC 1413*.

However, when an IPv6 address is assigned to the client host, the client is not checked by using the `identd` daemon even if `On` is specified. If %1 is specified as the log format, unknown is output to REMOTE_IDENT, which is a CGI environment valuable.

On: Checks the client by using the `identd` daemon

Off: Does not check the client by using the `identd` daemon

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>

(35) ImapBase {<u>map</u> | referrer | URL}

(a) Contents

This directive specifies the default base line of the image map file.

map: Location of map file

referrer: Location of document (Location of HTML file that displays the image map)

URL: Specified URL

For the URL, you can also specify an IPv6 address or the host name corresponding to an IPv6 address.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(36) ImapDefault {error | <u>nocontent</u> | map | referrer | URL}

(a) Contents

This directive specifies the default values for the default line of the image map file.

error: Displays the standard error message (The web server responds with the status code 500 Server Error).

nocontent: Ignores the request (The web server responds with the status code 204 No Content).

map: Displays the URL of map file in the menu.

referer: The web server responds with the status code 302 Found.

URL: Displays the contents of the specified URL.

For the URL, you can also specify an IPv6 address or the host name corresponding to an IPv6 address.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(37) ImapMenu {none | <u>formatted</u> | semiformatted | unformatted}

(a) Contents

The ImapMenu specifies the menu display when 'map' is provided in the specification value of the image map file or when the coordinates (0,0) on the image map are pointed with the mouse. The operation when an image map file is requested without any specified coordinates is also as per these settings.

none: Does not generate the menu. The operations are as per the default line specifications of the map file.

formatted: Displays the header and link list. The comments in the map file are ignored.

semiformatted: Displays the link list. Also, displays the comments in the map file.

unformatted: You can set the menu format independently by describing HTML in the map file.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(38) Include file-name

(a) Contents

The 'Include' directive makes the file specified in the file name available as the configuration file.

In the file name, you can specify the absolute path, or the relative path from the specified value of the ServerRoot directive. If there are multiple specifications of this directive, the merged contents are used. When the file contains the same directives, the directive specified later would overwrite the earlier ones.

(b) Location where you can code

httpsd.conf

(39) IndexIgnore file-name [file-name ...]

(a) Contents

The IndexIgnore directive specifies the file name that is not displayed on the Web browser when you display the directory index. You can specify even by the regular expressions.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t
```

(40) IndexOptions [{+ | -}]option [[{+ | -}]option ...]

(a) Contents

The IndexOptions directive sets the format display functionality options of the directory index. An option is enabled when + is specified before the option, or when `+-` is omitted. By default, all the options are disabled. The following table describes the list of options that you can specify:

Table 6-6: The Option list

Option	Meaning

`Charset=`*character-set* (ISO-8859-1 U (UTF-8 W	This option specifies the character set of the pages used for index display. If the character set used in the file specified in the HeaderName directive or the ReadmeName directive differs from the default character set (for the UNIX version: ISO-8859-1, for the Windows version: UTF-8), specify the same character set as for the file specified in the HeaderName directive or the ReadmeName directive. For this option, =*character-set* cannot be omitted. The operation when -Charset is specified is the same as the operation when +Charset is specified.
DescriptionWidth[={Number of characters | *}] <<23, 30, 42, or 49>>	This option specifies the width of file descriptive text area by number of characters (1 character=1 byte). If you specify *, the display is in accordance with the maximum length of the file descriptive text specified in the AddDescription directive. If you omit this option, the width of the file descriptive text area is 23 bytes (However, width is +7 when SuppressSize is specified and +19 when SuppressLastModified is specified). You can omit ={*number-of-characters* | *} when the -DescriptionWidth is specified. Display width in such case is 23 bytes.
FancyIndexing	Enables the format display functionality of the directory index.
FoldersFirst	Specify this option when performing the index display of the directory before the file. However, this is the case only when the FancyIndexing is enabled.
IconsAreLinks	Link the icons to files when you display the directory in index format.
IconHeight[=Number of pixels]((>0))<<22>>	This option specifies the height of icon in number of pixels when you display the directory index format. Specify this option along with the IconWidth option. This option becomes the HEIGHT attribute of the HTML IMG tag that displays the index.
IconWidth[=Number of pixels]((>0))<<20>>	This option specifies the icon width in pixels when you display the directory index format. Specify this option along with the IconHeight option. This option becomes the Width attribute of the HTML IMG tag that displays the index.
IgnoreCase	When you display the directory index format, this option sorts the file names and the directory names ignoring the case.
NameWidth[={*number-of-characters* | *}]<<23>>	This option specifies the width of file name and the directory name area with the number of characters (1 character=1byte). If you specify *, the display is in accordance with the maximum length of file name and the directory name. If you omit ={*number-of-characters* | *}, always specify this option as -NameWidth.

ScanHTMLTitles	When the AddDescription directive is not specified, search for the <TITLE> tag in the HTML file and display it as the descriptive text.
SuppressColumnSorting	This option disables the functionality that sorts the index into the columns of the file name, directory name, last updated date and time, file size, and file descriptive text.
SuppressDescription	This option does not display the descriptive text of the file.
SuppressHTMLPreamble	This option outputs both the contents of the file specified in the HeaderName directive and the HTML header that is automatically created (such as <HTML> and <TITLE>) when the HeaderName directive is specified. This option suppresses the output of the HTML header that is automatically created when the file specified in the HeaderName directive is written in HTML.
SuppressLastModified	This option does not display the last updated date and time.
SuppressSize	This option does not display the file size.
TrackModified	This option sets the Last-Modified value and Etag value in the HTTP response header of the response for directory display. If you specify this option, as the directory can check the file configuration changes, the client can use the cache functionality effectively. This option is valid only when the operating system and file system supports `stat()`.

(b) Notes

- When performing multiple specifications of this directory, you cannot specify different character strings for the same file name.
- When =*value* is specified by the IconHeight, IconWidth, and NameWidth, you cannot specify the - option
- The set options are inherited from the upper directory to the lower directory in the order of httpsd.conf, <VirtualHost>, <Directory>, and .htaccess. Merge the inherited options eventually and determine the format for index display.
- The options are disabled even when you add +- in httpsd.conf and specify the options. However, the options are inherited to the lower directory in the order of httpsd.conf, <VirtualHost>, <Directory>, and .htaccess. The merging process enables the specifications of the inherited options. When the options for which the reference order is in the lower locations are specified or if any of the following directives are specified, the merging process is executed:
    - AddAlt
    - AddAltByEncoding

    - AddAltByType
    - AddDescription
    - AddIcon
    - AddIconByEncoding
    - AddIconByType
    - DefaultIcon
    - HeaderName
    - ReadmeName

  (Example)

  When IndexOptions +FancyIndexing +IconsAreLinks are specified in the httpsd.conf file, and if the index related directives are not specified in the lower specification location, the FancyIndexing and IconsAreLinks are disabled.

  When the IndexOptions +FancyIndexing +IconsAreLinks are specified in the httpsd.conf, and AddDescription "text file" *.txt is specified in the access control file of the lower directory, FancyIndexing and IconsAreLinks are enabled.

- If you specify Charset, IconHeight, IconWidth, and NameWidth directives without +- specifications, the options with +- specification (excluding Charset, IIconHeight, IconWidth, NameWidth) that are specified before specifying these options in the specified location are disabled.

  (Example)

  IndexOptions FancyIndexing -IconsAreLinks IconHeight IconWidth

  In this case, the FancyIndexing, IconHeight, and the IconWidth directives are enabled. The - specification of IconsAreLinks is not inherited.

- In the merging process where options for the same directory index are specified between the specified locations, if you specify options without +- at the rear location in the reference order, the options specified earlier are disabled. However, IconHeight, IconWidth, and NameWidth are not disabled.

  (Example 1)

    - Specification of the httpsd.conf file

      IndexOptions + FancyIndexing + IconsAreLinks

    - Specification of the access control file

      IndexOptions FancyIndexing SuppressLastModified

  When you specify these options, IconsAreLinks are disabled. FancyIndexing and SuppressLastModified are enabled.

(Example 2)

- Specification of the httpsd.conf file

  IndexOptions SuppressColumnSorting + FancyIndexing + IconsAreLinks
- Specification of the access control file

  IndexOptions FancyIndexing SuppressLastModified

If you specify these options, SuppressColumnSorting and IconsAreLinks are disabled. FancyIndexing and SuppressLastModified are enabled.

- In the merging process where options for the same directory index are specified between the specified locations, if you specify both the + and - for the same options, - specification is enabled.

  (Example)

  - Specification of the httpsd.conf file

    IndexOptions + FancyIndexing - IconsAreLinks
  - Specification of the access control file

    IndexOptions + IconsAreLinks

  If you specify these options, IconsAreLinks are disabled.
- If you specify the option in which +- is not specified in the same specification location, the options other than the Charset, IconHeight, IconWidth, and the NameWidth directive, specified with +- are disabled.

  (Example 1)

  - Specification of httpsd.conf file

    IndexOptions + IconsAreLinks FancyIndexing + SuppressLastModified

  In this case, IconsAreLinks is disabled.

  (Example 2)

  - Specification of <VirtualHost> block, <Directory> block, or access control file.

    IndexOptions + IconsAreLinks FancyIndexing + SuppressLastModified

  In this case, IconsAreLinks and SuppressLastModified are disabled.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(d) Overwrite permission

Indexes level

(41) IndexOrderDefault {Ascending | Descending} {Name | Date | Size | Description}

(a) Contents

The IndexOrderDefault directive specifies the default sorting order for the files in the directory index display.

Ascending: Ascending order

Descending: Descending order

Name: Sort by the file name.

Date: Sort by the file update date.

Size: Sort by the file size.

Description: Sort by the descriptive text specified in the AddDescription directive.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

6.2.5 Directives starting with K and L

(1) KeepAlive {On | Off}

(a) Contents

The KeepAlive directive specifies whether to enable the KeepAlive connection. Actually the KeepAlive is executed only when the client also supports the KeepAlive. As the KeepAlive keeps the persistent connection between server process and the client, the response to continuous requests is good. On the other hand, as a server process is exclusively reserved for specific clients, the service efficiency of the Web server as a whole may decline. You need to make adjustments by using the KeepAliveTimeout and the MaxKeepAliveRequests directives.

On: Enables the persistent connection (KeepAlive).

Off: Disables the persistent connection (KeepAlive).

(b) Location where you can code

httpsd.conf

(c) Specification example

```
KeepAlive On
```

(2) KeepAliveTimeout time

~((0 - 65535))<<15>> (Unit: Seconds)

(a) Contents

This directive specifies the request waiting time during the KeepAlive connection in seconds. If the request waiting time elapses and the next request does not come from the client, the connection is disconnected. In the KeepAlive connection, persistent clients occupy the server process. Specify the settings in such a way that when the standard time required for moving from one Web page to the next Web page is exceeded, the timeout disconnects the connection and the server process is applied for processing other request. If the time is set to `0`, the KeepAlive connection becomes invalid.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
KeepAliveTimeout 15
```

The request waiting time is 15 seconds in the case of KeepAlive connection.

(3) LanguagePriority language-code [language-code ...]

(a) Contents

The LanguagePriority directive specifies the used languages in the order of descending priorities. In the content negotiation, if the priority order (Accept-Language header) of the language code is not included in the request from the Web browser, the specified priority order is used. For the language code specified here, see *AddLanguage directive*.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
LanguagePriority ja en fr de
```

The priority order is Japanese, English, French, and German.

(4) LDAPBaseDN DN-value [DN-value...]

(a) Contents

When you authenticate in the LDAP server, the LDAPBaseDN directive specifies the uppermost DN to start search. With this DN, you need to permit the access from the

Web server.

All the user entries and group entries to be searched must be present under this DN. The Web server requires access rights to all these entries and attribute. When multiple LDAP servers are specified with the LDAPServerName directive, specify DN for each LDAP server. When there are multiple specifications of DN, enclose each DN with " (double quotation marks). If only one DN is specified, you need not enclose the DN with quotation marks. If DN includes quotation marks, add a ¥ before the quotation mark.

(b) Note

The mod_hws_ldap module must be embedded for user authentication via the LDAP server. For details on user authentication via the LDAP server, see *4.5.4 User authentication and access control using the directory service*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>

(5) LDAPNoEntryStatus {<u>Authorization</u> | Forbidden}

(a) Contents

This directive specifies the status code that the Web server returns to the Web browser when the access is denied by the access control of the LDAPRequire directive, even if the user authentication with the LDAP server is successful.

Authorization: Returns the status code 401.

Forbidden: Returns the status code 403.

(b) Note

The mod_hws_ldap module must be embedded for user authentication via the LDAP server. For details on user authentication via the LDAP server, see *4.5.4 User authentication and access control using the directory service*.

(c) Location where you can code

httpsd.conf, <VirtualHost>

(6) LDAPRequire [% DN-attribute %] [LDAP-search-filter]

(a) Contents

Specify the LDAPRequire directive with the AuthName directive, AuthType directive, and the Require valid-user directive and specify the range of users for whom access is to be controlled.

If there is a character string enclosed by % in the beginning, this character string is used as the DN attribute that identifies the user name entered by the client. If the character string does not contain %, the Web Server assumes cn (mandatory attribute that identifies a person defined on the basis of entry) as the DN attribute.

The DN registered by the user is requested from the specification value of the LDAPBaseDN directive by combining the user name entered by the client and a value set as the DN attribute. Use this DN and the password entered by the client to perform the authentication in the LDAP server.

When the specified user is authenticated in LDAP server and is applicable to the LDAP search field, the user can access contents. When filter is not specified, (objectClass=*) is specified as search filter.

When this directive is simultaneously used with SSL client authentication, the access to the LDAP server becomes the anonymous access without any password and with the CN value of the Subject field in the client certificate as the user name. As a result of the LDAP server search, if the user name is registered in the LDAP server and is applicable to the LDAP search filter, the user can access the contents. If the filter is not specified, (objectClass=*) is set as a search filter.

When you specify this directive, the specification value of the Require directive specified in the directive will be disabled. However, the specification of the Require directive is mandatory.

Specify the search filter in the following format:

(Attribute Operator Value)

The following table describes the operators that you can use:

Table 6-7: Operators that you can use in search filter

Search type	Symbol	Description
Equality	=	Returns the entry containing the attribute entry set in the specification value. **Example:** cn=hitachi taro
Substring	=<Character string>*<Character string>	Returns the entry containing the attribute that includes the specified sub-character string. **Example:** cn=hita*, cn=*hanako, cn=*hi*, and cn=h*hanako
Greater than or equal to	>=	Returns the entry that includes the attributes greater than or equal to the specified value. **Example:** employeenumber>=100
Less than or equal to	<=	Returns the entry that includes the attributes less than or equal to the specified value. **Example:** employeenumber<=100

Search type	Symbol	Description
Presence	=*	Returns the entry that includes the specified attribute. **Example:** cn=*, telephonenumber=*, and manager=*

Moreover, you can create a filter by combining two or more of these search filters.

(Operator (Search filter)(Search filter) ...)

In this case, you can use the operators described in the following table.

If the authenticated user has multiple attribute entries for one attribute, access is granted if one attribute entry matches with the operation.

Table 6-8: Operators that you can use in multiple search filters

Operators	Symbol	Description
And	&	All the filters return true entry. **Example:** (&(filter)(filter)(filter) ...)
Or	|	Atleast one specified filter returns true entry. **Example:** (|(filter)(filter)(filter) ...)
Not	!	Specified filter returns the entry that is not true. **Example:** (!(filter))

# You cannot specify multiple filters in the case of Not operator.

The Web server responds with the status code 401 Authorization Required, if the authentication fails. When the entry does not match with the filter condition, the Web server responds with the status code (by default 401 Authorization Required) as per the LDAPNoEntryStatus directive. When the filter format syntax is wrong, the status code 500 Internal Server Error is returned.

If there are multiple specifications of the same user, the access is possible if access permission is granted to one of the specified users.

(b) Note

The mod_hws_ldap module must be embedded for user authentication via the LDAP server. For details on user authentication via the LDAP server, see *4.5.4 User authentication and access control using the directory service*.

(c) Location where you can code

<Directory>, .htaccess

(d) Overwrite permission

AuthConfig level

(e) Specification example

- When access permission is granted to the user name hitachi taro and hitachi hanako
  `LDAPRequire (|(cn=hitachi taro)(cn=hitachi hanako))`

- When access permission is granted to user IDs from 99001 to 99029, and to the user IDs from 99051 to 99059
  ```
  LDAPRequire %uid%
  (|(&(uid>=99001)(uid<=99029))(&(uid>=99051)(uid<=99059)))
  ```

(7) LDAPServerName {host-name | IP-address} [{host-name | IP-address}...]

~<<127.0.0.1>>

(a) Contents

The LDAPServerName directive specifies the host name or the IP address of the LDAP server. When you specify multiple LDAP servers, specify the respective port numbers for each LDAP server (LDAPServerPort directive) and the upper-most DN (LDAPBaseDN directive) that starts the search. If the LDAP servers, the port numbers, and the number of DNs do not match, the authentication is not executed on the LDAP server. The Web Server outputs error log and returns the status code 500 to the Web browser.

However, when you use the same port number for all the LDAP servers specified in this directive, you can simplify the port number specification.

When you specify multiple LDAP servers, specify in the order descending priorities. Always perform the user authentication for each request, from the LDAP server that was specified in the beginning.

Local host (127.0.0.1) is set by default.

Note that a host name corresponding to an IPv6 address cannot be specified for *host-name*. An IPv6 address also cannot be specified for *IP-address*.

(b) Note

The mod_hws_ldap module must be embedded for user authentication via the LDAP server. For details on user authentication via the LDAP server, see *4.5.4 User authentication and access control using the directory service*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>

(d) Specification example

```
LDAPServerName server01 server02 server03
LDAPServerPort 389
LDAPBaseDN "ou=employee, o=hitachi, c=jp" "ou=employee,
o=example.com" "o=hitachi, c=jp"
```

If you specify as mentioned above, the combination of the LDAP server, port number, and the DN is as follows:

Order of the LDAP server to be used	Host name	Port number	DN that starts search
1	server01	389	ou=employee, o=hitachi, c=jp
2	server02	389	ou=employee, o=example.com
3	server03	389	o=hitachi, c=jp

(8) LDAPServerPort port-number [port-number...]

~((1 - 65535))<<389>>

(a) Contents

The LDAPServerPort directive specifies the LDAP server port number. When multiple LDAP servers are specified in the LDAPServerName directive, specify the port number for each LDAP server. However, when you use the same port number for all the LDAP servers specified in the LDAPServerName directive, you need not specify the number multiple times. Specify only one port number that applies to all the LDAP servers. For the example of multiple LDAP server specifications, see *LDAPServerName directive*. If the number of LDAP servers and the port numbers do not match, an error occurs.

(b) Note

The mod_hws_ldap module must be embedded for user authentication via the LDAP server. For details on user authentication via the LDAP server, see *4.5.4 User authentication and access control using the directory service*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>

(9) LDAPSetEnv environment-variable attribute

(a) Contents

When you authenticate with the LDAP server, this directive sets the value of the attribute that configures the entries identified by the DN of authenticated users, as the value of the environment variable. Set the attribute value with the character code acquired from the LDAP server. For the details on the returned character codes, see the

LDAP server manual. When there are multiple values for one attribute, the environment variable is not set. You cannot specify the attribute with a binary option (;binary). For the attributes containing other options, the attribute value is not set properly unless the value is not a character string. When the same environment variable is defined in another attribute, error does not occur but the value of environment variable becomes undefined.

(b) Note

The mod_hws_ldap module must be embedded for user authentication via the LDAP server. For details on user authentication via the LDAP server, see *4.5.4 User authentication and access control using the directory service*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(d) Overwrite permission

FileInfo level

(10) LDAPTimeout value

~((1 - 86400))<<30>> (Unit: Seconds)

(a) Contents

After the user authentication, the LDAPTimeout directive specifies the maximum waiting time in seconds for the search process of the filter specified in one LDAPRequire directive. The status code 500 Internal Server is returned in any of the following cases:

- When there is no response within the period specified in this directive
- When timeout is notified by the LDAP server itself
- When the LDAP server fails to access

(b) Note

The mod_hws_ldap module must be embedded for user authentication via the LDAP server. For details on user authentication via the LDAP server, see *4.5.4 User authentication and access control using the directory service*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>

(11) LDAPUnsetEnv environment-variable

(a) Contents

This directive disables the environment variables specified in the LDAPSetEnv directive.

(b) Note

The mod_hws_ldap module must be embedded for user authentication via the LDAP server. For details on user authentication via the LDAP server, see *4.5.4 User authentication and access control using the directory service*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(d) Overwrite permission

FileInfo level

(12) LimitRequestBody ***request-body-size***

~((0 - 2147483647))<<0>> (Unit: bytes)

(a) Contents

This directive specifies the upper limit for the object body (data) size when the server receives a request from the Web browser using the HTTP communication. The object body is used when the request is sent by <FORM METHOD=POST ACTION=...> from the Web server. When the upper limit is not specified, it is set to `0`.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(13) LimitRequestFields ***number-of-headers***

~((0 - 32767))<<100>>

(a) Contents

This directive specifies the upper limit for the number of HTTP headers when the server receives a request from the Web browser using the HTTP communication. The number of HTTP headers of the request changes as per the specification of the proxy that links the Web browser and requests. When the upper limit is not specified, it is set to `0`.

(b) Location where you can code

httpsd.conf

(14) LimitRequestFieldsize ***header-size***

~((0 - 8190))<<8190>> (Unit: bytes)

(a) Contents

This directive specifies the upper limit for the size of one HTTP header when the server receives the request from the Web browser using the HTTP communication. The size of request header changes as per the specification of the proxy that links the Web browser and requests.

(b) Location where you can code

httpsd.conf

(15) LimitRequestLine request-line-length

~((0 - 8190))<<8190>> (Unit: bytes)

(a) Contents

This directive specifies the upper limit for the length of the request string (including the URI, HTTP version, the method and the inquiry character string) when the server receives the request from the Web browser using the HTTP communication. When the request is sent from the Web browser by <FORM METHOD=GET ACTION...>, the request string is used as a query string. Note that the number of bytes sent from the Web browser as the request line changes as per the specification of the proxy that links the Web browser and requests.

(b) Location where you can code

httpsd.conf

(16) Listen [IP-address:] port-number

(a) Contents

The 'Listen' directive specifies the IP address and the port number that receives the request. Unlike 'Port' directive, you can perform multiple specifications. Specify this directive when defining the virtual host. When you specify the Listen directive, the specifications of Port directive and BindAddress directive are ignored.

You can specify an IPv6 address for *IP-address*. Specify an IPv6 address by enclosing it in square brackets ([ ]). However, if you omit *IP-address* and specify only the port number, only requests using IPv4 addresses are accepted. Therefore, when using an IPv6 address, be sure to specify the IPv6 address in the Listen directive.

To restart the server after changing the IP address specified in the Listen directive, stop the server, and then start it. If you use other means to restart the server, such as a command, startup might fail.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
Listen 80
Listen [2001::123:4567:89ab:cdef]:8080

Listen [::]:80
```

(17) ListenBacklog number-of-backlogs

~((1 - 2147483647))<<511>>

(a) Contents

The ListenBacklog directive specifies the maximum queue size for the connection requests from the client. The specified value is set as the number of backlogs of system call listen( ). However, as the limit for the specification value and the actual maximum value for queue size differ according to the OS, see the OS manual for listen( ) and documents that explain TCP/IP implementation for each OS.

(b) Location where you can code

httpsd.conf

(18) LoadFile file-name [file-name ...]

(a) Contents

The LoadFile directive specifies the object file or the library containing the codes that are referred by the module incorporated by the DSO. In the file name, you can specify the absolute path, or the relative path from the specified value of the ServerRoot directive.

When you specify the modules that refer to this file in the LoadModule directive, you need to specify this directive before these modules are used in httpsd.conf.

(b) Location where you can code

httpsd.conf

(19) LoadModule module-structure-name library-file-name

(a) Contents

This directive specifies a module to be dynamically embedded in the Web server. You can specify absolute path, or the relative path from the specified value of the ServerRoot directive in the library file name.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
LoadModule hws01_module libexec/mod_hws01.so
LoadModule hws02_module libexec/mod_hws02.so
```

Embed the module hws01_module and the module hws02_module.

(20) LogFormat "format" [label-name]

~<<"%h %l %u %t ¥ "%r ¥ " %>s %b">>

(a) Contents

The LogFormat directive defines the label name in log format. You can specify the label name defined here in the CustomLog directive. For the format that you can specify, see *CustomLog directive*. If label name is not attached, you cannot specify this directive multiple times. If %A or %a is specified in the format, IPv6 addresses can also be output. If %h or %V is specified in the format, host names corresponding to IPv6 addresses or the IPv6 addresses can also be output.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
LogFormat "%h %l %u %t ￥"%r￥" %>s %b ￥"%{Referer}i￥"
￥"%{User-Agent}i￥"" combined
LogFormat "%h %l %u %t ￥"%r￥" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
```

(21) LogLevel {debug | info | notice | warn | error | crit | alert | emerg}

(a) Contents

This directive specifies the level of the errors that are output to the error log. The Web Server outputs the upper level error log than the specified level. Note that notice level logs are output regardless of this specification. Messages that are output before the analysis of the level specification finishes (for example, during Cosminexus HTTP Server startup) may be output regardless of this specification.

The following table describes the error levels in the ascending order:

Level	Meaning
emerg	Emergency message
alert	Message that requests instant processing
crit	Critical state message
error	General error message
warn	Warning level message
notice	Standard but important message
info	Information messages, and module trace information# collected when external modules and CGI programs are executed
debug	Debug level messages, trace information for internal modules, and info-level module trace information#

#: You can specify that module trace information is to be output not to the error log but to the request log. For details, see *4.2.2 (5) Locations to which trace information is* *output* and *4.2.6 Collecting the module trace*.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
LogLevel info
```

6.2.6 Directives starting with M, N, O, P, Q, and R

(1) MaxClients number-of-connections U

~((1 - 1024)) <<1024>>

(a) Contents

The MaxClients directive specifies the maximum number of clients that can be connected simultaneously.

When you start the server, the processes equivalent to the number specified in the StartServers directive start and wait for requests. When many requests occur simultaneously, multiple processes process these requests. When the remaining number of processes waiting for request is less than the number specified in the MinSpareServers directive, new processes are gradually generated. The processes are generated until the number of total processes is equal to the number specified in the directive. After that, if the request processing ends and the request waiting processes increase, the processes are terminated until the number of processes specified in the MaxSpareServers directive is reached.

For details on the other directives related to number of processes, see *4.1 Relationship between processes and directives of Cosminexus HTTP Server*.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
MaxClients 150
```

(2) MaxKeepAliveRequests number-of-connections

~((0 - 2147483647))<<100>>

(a) Contents

This directive specifies the upper limit for the number of KeepAlive persistent connections. If the upper limit is not specified, it is set to 0. In KeepAlive, since specific clients occupy the server processes, and hence set the upper limit to give chance to other clients also to receive the services.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
MaxKeepAliveRequests 100
```

(3) MaxRequestsPerChild request-processing-frequency U

~((0-2147483647)) <<0>>

(a) Contents

This directive specifies the request processing frequency of a server process. A server process ends after processing the specified number of requests. This has an advantage to prevent the failure due to memory leakage by the user-created applications. If the frequency is set to `0` the upper limit for the request processing frequency of a server process is not set. In such cases the server process does not end and waits for processing the next request.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
MaxRequestsPerChild 10000
```

(4) MaxSpareServers number-of-processes U

~((1 - 1024))<<10>>

(a) Contents

This directive specifies the maximum number of server processes that are running in the request waiting state. For details on other directives related to number of processes, see *4.1 Relationship between processes and directives of Cosminexus HTTP Server*.

If the value of this directive is less than the value of the MinSpareServers directive, a value of the MinSpareServers specification value + 1 is assumed.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
MaxSpareServers 10
```

(5) MinSpareServers number-of-processes U

~((1 - 1024))<<5>>

(a) Contents

This directive specifies the minimum number of server processes that are running in the request waiting state. If number of server processes is less than the specified value, the Web server generates new processes. For details on other directives related to number of processes, see *4.1 Relationship between processes and directives of Cosminexus HTTP Server*.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
MinSpareServers 5
```

(6) MultiviewsMatch {NegotiatedOnly | Handlers}

(a) Contents

The MultiviewsMatch directive specifies the extension types that are the targets of content negotiation.

NegotiatedOnly: Extensions that are related only to character sets, compression format, language code, and MIME type are the targets of content negotiation.

Handlers: In addition to the extensions specified in NegotiatedOnly, the extensions related to the handler are also the target of content negotiation.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
MultiviewsMatch Handlers
```

*(7) NameVirtualHost {IP-address | *}[:Port-number]*

(a) Contents

This directive specifies the IP address that is used by the virtual host based on the sever name. An IPv6 address can be specified for *IP-address*. Specify an IPv6 address by enclosing it in square brackets ([ ]).

If you specify an asterisk (*) instead of an IP address, a virtual host based on the server name is generated for connections from IPv4 addresses other than the IPv4 addresses in both the NameVirtualHost directive that specifies IPv4 addresses and in the <VirtualHost> block. This specification is convenient when you use only a virtual host based on server name and do not want to set IP address in configuration file.

(b) Location where you can code

httpsd.conf

(8) Options {+ | -} options [{+ | -} options ...]

~<<All>>

(a) Contents

Specify this directive for restricting the functionality that the user can use.

+: Permits the use of functionality specified in the option.

-: Prohibits the use of functionality specified in the option.

Option	Functionality
All	All the options excluding MultiViews and SymLinksIfOwnerMatch are enabled.
ExecCGI	Permits the execution of the CGI script.
FollowSymLinks	Traces the symbolic link. You cannot specify this option in Windows version.
Indexes	When the directory is specified in the URL, and if the file (by default index.html) specified in the DirectoryIndex directive does not exist, this option displays the directory index.
MultiViews	Supports the Content-negotiated Multiviews.
None	Disables the functionality that you can specify for all options.
SymLinksIfOwnerMatch	Traces the link only when the owner of file or directory and the owner of symbolic link are same. You cannot specify this option in Windows version.

# If you specify the directive multiple times without using +-, only the last specified directive is enabled.

(Example 1)
```
Options All
Options ExecCGI
```

As shown in this example, if you specify the directives in two lines without specifying the +- in the option, the user can use only the execution functionality of the CGI script. The functionality such as directory index functionality cannot be used.

(Example 2)

Specification of the httpsd.conf file
```
    Options All
```

Specification of the access control file

```
    Options ExecCGI
```

The access control file is referred to after the httpsd.conf file, and hence you can use only the execution functionality of the CGI script in the directory with the access control file.

(Example 3)

```
    Options Indexes ExecCGI
```

As shown in this example, if you specify the options in one line without +- signs, you can use both the specified functionality.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Options level

(9) Order directives

~<<deny,allow>>

(a) Contents

This directive specifies the evaluation order of the Allow directive and the Deny directive. You can specify the following in the directive. Earlier evaluated directives are overwritten by the later ones.

Directives	Meaning
deny,allow	Evaluates the specification of Deny directive before the specification of Allow directive.
allow,deny	Evaluates the specification of Allow directive before the specification of Deny directive.
mutual-failure	Permits the access only to the host where the Allow directive is specified and Deny directive is not specified.

(b) Location where you can code

<Directory>, .htaccess

(c) Overwrite permission

Limit level

(10) PassEnv environment-variable [environment-variable ...]

(a) Contents

You can specify an optional environment variable to pass on to the CGI script.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
PassEnv TMP
```

(11) PidFile file-name

~<<logs/httpd.pid>>

(a) Contents

This directive specifies a file name that stores control process ID. In file name, you can specify absolute path, or the relative path from the specified value of ServerRoot directive.

(b) Note

In Windows, changes to the values specified in the PidFile directive are not applied when you restart the server. If you have changed values specified in the PidFile directive, stop and then restart the Web server.

In UNIX, changes to the values specified in the PidFile directive are not applied when you restart the server. If you have changed values specified in the PidFile directive, use the `kill` command to stop the Web server, and then restart the Web server. The httpsdctl command cannot be used while the Web server is stopped.

(c) Location where you can code

httpsd.conf

(d) Specification example

```
PidFile logs/httpd.pid
```

(12) Port port-number

~((1 - 65535))<<80>>

(a) Contents

The Port directive specifies the port number of the server that receives requests from Web browsers that use IPv4 addresses.

Requests from Web browsers that use IPv6 addresses are not received even if the Port directive has been specified. To use IPv6 addresses, specify the IPv6 addresses in the Listen directive. In addition, if you are simultaneously using IPv4 addresses, also specify the IPv4 address in the Listen directive.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
Port 80
```

(13) ProxyErrorOverride {On | Off}

(a) Contents

The ProxyErrorOverride directive overrides the response header and response body when the backend server returns a three-digit status code beginning with 3, 4, or 5. As a result, the reverse proxy sends a self-generated response to the client instead of the response from the backend server.

On: Overrides the response header and response body when the backend server returns a three-digit status code beginning with 3, 4, or 5.

Off: Does not override the response header or the response body.

(b) Note

The mod_proxy and mod_proxy_http modules must be embedded to use a reverse proxy. For details on reverse proxies, see *4.7 Setting the reverse proxy*.

(c) Location where you can code

httpsd.conf, <VirtualHost>

(d) Specification example

```
ProxyErrorOverride On
```

When the backend server returns a three-digit status code beginning with 3, 4, or 5, the response generated by the reverse proxy is returned to the client.

(14) ProxyPass path-name URL

(a) Contents

When using a reverse proxy server, the ProxyPass directive specifies the request received from the Web browser and the address used to forward the request.

path-name: Specifies the request from the Web browser to the reverse proxy as a URL that starts with a forward slash (/).

URL: Specifies the backend server URL to which the request is forwarded in the format http://*host-name*[:*port-number*].

You can also specify an IPv6 address or the host name corresponding to an IPv6 address for *URL*.

You cannot specify a path name that is a duplicate of any of the following directive specification values:

- JkMount URL patterns in the redirector definition file

(b) Note

The mod_proxy and mod_proxy_http modules must be embedded to use a reverse proxy. For details on reverse proxies, see *4.7 Setting the reverse proxy*.

(c) Location where you can code

httpsd.conf, and <VirtualHost>

(15) ProxyPassReverse path-name URL

(a) Contents

When using a reverse proxy server, the ProxyPassReverse directive changes the URL coded in the Location header field in the redirect response sent from the backend server. To make the redirect request from the Web browser pass through the reverse proxy server, change the Location header to the value specified in this directive.

path-name: Specifies the path of the reverse proxy that receives the redirect request as a URL that starts with a forward slash (/).

URL: Specifies the backend server URL in the Location header that is to be changed, in the format http://*host-name*[:*port-number*].

You can specify an IPv6 address or the host name corresponding to an IPv6 address for *URL*. IPv6 addresses have various formats, so use care when specifying values. If the backend server URL differs from the specified value, the directive is not enabled. When specifying an IPv6 address, check the format of the IPv6 address in the Location response header that is sent from the backend server.

(b) Note

The mod_proxy and mod_proxy_http modules must be embedded to use a reverse proxy. For details on reverse proxies, see *4.7 Setting the reverse proxy*.

(c) Location where you can code

httpsd.conf, <VirtualHost>

(16) ProxyPreserveHost {On | <u>Off</u>}

(a) Contents

When using a reverse proxy server, the ProxyPreserveHost directive specifies whether to forward the Host header value received from the client to the backend server without changes.

On: Forwards the Host header value received from the client to the backend server without changes.

Off: Changes the Host header value received from the client according to the value specified in the ProxyPass directive, and then forwards the changed value to the backend server.

(b) Note

The mod_proxy and mod_proxy_http modules must be embedded to use a reverse proxy. For details on reverse proxies, see *4.7 Setting the reverse proxy*.

(c) Location where you can code

httpsd.conf, <VirtualHost>

(d) Specification example

```
ProxyPreserveHost On
```

The Host header value received from the client is forwarded to the backend server without changes.

(17) ProxyVia {on | <u>off</u> | full | block}

(a) Contents

Specify this directive when you control the use of Via header with the proxy.

on: Adds the information of the local host to the Via header. The existing information does not change.

off: Does not add the information of the local host to the Via header. The existing information does not change.

full: Adds the information in which version of the local host is added as comment, to the Via header. The existing information does not change.

block: Does not add the information of the local host to Via header. Deletes the Via header in the request.

(b) Note

The mod_proxy and mod_proxy_http modules must be embedded to use a reverse proxy. For details on reverse proxies, see *4.7 Setting the reverse proxy*.

(c) Location where you can code

httpsd.conf, <VirtualHost>

(18) QOSCookieDomain domain-name

(a) Contents

The QOSCookieDomain directive specifies the domain where the cookies used in flow control functionality is enabled. This value is used in HWS creation mode, not in user creation mode. When multiple hosts are set, by specifying this directive you can use cookies between the hosts that share the domain. In the domain name at least two "." must be included.

Note that you can specify the domain name corresponding to an IPv6 address.

(Example)

When two hosts, a.example.com and b.example.com are set, if .example.com is specified in this directive, priority processing is performed even when you access any of the two hosts.

(b) Note

The mod_hws_qos module must be embedded to use flow-restricting functionality. For details on flow-restricting functionality, see *4.9 Flow-restricting functionality*.

(c) Location where you can code

httpsd.conf, and <VirtualHost>

(19) QOSCookieExpires value

~((0 - 86400))<<300>> (Unit: Seconds)

(a) Contents

This directive specifies the validity period of the cookie that is used in the flow restriction functionality in seconds. This value is used in HWS creation mode, not in user creation mode.

(b) Note

The mod_hws_qos module must be embedded to use flow-restricting functionality. For details on flow-restricting functionality, see *4.9 Flow-restricting functionality*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Location>

(20) QOSCookieName cookie-name [{hws | user}]

~<<HWSCHK>>

(a) Contents

The QOSCookieName directive specifies the name of the cookie to be used in flow-restriction functionality. Semicolons (;), commas (,), and spaces cannot be used in the cookie name. For session management that uses different cookies between the host and the URL, a different name must be specified.

hws: Manages sessions by using cookies generated by Cosminexus HTTP Server. This is called *HWS creation mode*.

user: Manages sessions by using cookies generated in an external module other than Cosminexus HTTP Server. This is called *user creation mode*.

(b) Notes

- The mod_hws_qos module must be embedded to use flow-restricting functionality. For details on flow-restricting functionality, see *4.9 Flow-restricting functionality*.
- If the QOSCookieName directive is specified in a specific block, the QOSCookieName directive that is specified in a higher location is not inherited.

  (Example)

  ```
  QOSCookieName Cookie1 hws
  <Location /loc1>
  QOSCookieName Cookie2 user
  </Location>
  ```

  In this case, if a request starts with /loc1, the specification of the cookie name Cookie2 is valid. If a request does not start with /loc1, the specification of the cookie name Cookie1 is valid.
- Do not duplicate cookie names when specifying the QOSCookieName directive multiple times. If cookie names are duplicated, a startup error occurs.

  (Example)

  ```
  QOSCookieName Cookie1 hws
  QOSCookieName Cookie1 user
  ```

  In this case, a startup error occurs because of cookie name duplication.
- If you specify the QOSCookieName directive in HWS creation mode multiple times, the last specification is valid.

  (Example)

  ```
  QOSCookieName Cookie1 hws
  QOSCookieName Cookie2 hws
  ```

  In this case, the specification of the cookie name Cookie1 is invalid, and the specification of the cookie name Cookie2 is valid.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Location>

(21) QOSCookieSecure {on | off}

(a) Contents

The QOSCookieSecure directive specifies the setting such that cookies are sent to the client only during SSL access. This value is used in HWS creation mode, not in user creation mode. Note that the cookie is confirmed after the encryption process of SSL ends.

on: Specifies the setting such that the cookies are sent to the client only during SSL access.

off: Specifies the setting such that the cookies are sent to the client even when the access is not with SSL.

(Example)

If you set this directive when the hosts with enabled SSL and the hosts with disabled SSL are set, the cookies are sent only when accessing the hosts with enabled SSL.

(b) Note

The mod_hws_qos module must be embedded to use flow-restricting functionality. For details on flow-restricting functionality, see *4.9 Flow-restricting functionality*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Location>

(22) QOSCookieServers value

In UNIX version

~((0 - MaxClients directive specification value))<<10>>

In Windows version

~((0 - ThreadsPerChild directive specification value))<<10>>

(a) Contents

When the number of server processes in the request waiting state is decreased, this directive specifies the number of server processes used when processing only the requests that are received by sending cookies.

In Windows version, specify the number of server threads.

(b) Note

The mod_hws_qos module must be embedded to use flow-restricting functionality. For details on flow-restricting functionality, see *4.9 Flow-restricting functionality*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Location>

(23) QOSRedirect old-path new-path

(a) Contents

Specify this directive when a process is denied by the flow control functionality, and the request from client is to be redirected to the specified path. In the new path, specify the URL path that includes "Protocol name://host name[:port number]".

You can also specify an IPv6 address or the host name corresponding to an IPv6 address for *new-path*.

When a request is redirected to the old path, the server returns a response that contains status code 302 and the Location header with the new path. The response cannot be customized.

For details on the specification of the old path and new path, see *Redirect directive*.

(b) Note

The mod_hws_qos module must be embedded to use flow-restricting functionality. For details on flow-restricting functionality, see *4.9 Flow-restricting functionality*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Location>

(24) QOSRejectionServers value

In UNIX version

~((0 - MaxClients directive specification value))<<1>>

In Windows version

~((0 - ThreadsPerChild directive specification value))<<1>>

(a) Contents

This directive specifies the value when the number of the server processes in the request waiting state is decreased and all the received requests are denied.

In Windows version, specify the number of server threads.

(b) Note

The mod_hws_qos module must be embedded to use flow-restricting functionality. For details on flow-restricting functionality, see *4.9 Flow-restricting functionality*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Location>

(25) QOSResponse {file[MIME-type] file-name | message text}

(a) Contents

When the process is denied by the flow control functionality, this directive specifies the contents to be returned along with the status code 503. The contents are cached in the server process, so you need to restart the server whenever there is a change.

file: Returns the specified file with the specified MIME type. When you omit the MIME type, "text/html" is set. In the file name, you can specify absolute path or the relative path from the specified value of the ServerRoot directive.

message: Returns the specified text. Specify the character string with quotation mark (") at the beginning for the text. In the MIME type, "text/html" is set.

(b) Note

The mod_hws_qos module must be embedded to use flow-restricting functionality. For details on flow-restricting functionality, see *4.9 Flow-restricting functionality*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Location>

(d) Specification example

```
QOSResponse file "text/html; charset=ISO-8859-1" htdocs/
busy.html
QOSResponse message "Server busy.
```

(26) ReadmeName file-name

(a) Contents

The ReadmeName directive specifies the file name (without the path information) of the file that describes the comment added as Readme, when the directory index is displayed. You can describe the comment in the HTML or plain text format. However, in the file specified with the AddType directive or the TypesConfig directive, the MIME type must be defined correctly. When you create command in plain text, <PRE> tag is added to the HTML of directory index display.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

Indexes level

(d) Specification example

```
ReadmeName README.html
```

Display the contents of README.html file under the directory.

(27) Redirect [{permanent | <u>temp</u> | seeother | gone | status code}] old-path new-path

(a) Contents

Specify the directive when the request from the client for old path is (redirected) to the new path.

In the old path, specify the request URL path that starts with a slash. However, you cannot specify the characters after the `?` sign (query string) in the old path.

You cannot specify an old path that is a duplicate of any of the following directive specification values:

- ProxyPass path name
- JkMount URL patterns in the redirector definition file

For example, the following paths cannot be specified:

```
Redirect temp /aaa/bbb/ http://aaa.example.com/
ProxyPass /aaa/ http://aaa.example.com/
```

In the new path, specify the URL path that includes "protocol name://host name[:port number]". You can also specify an IPv6 address or the host name corresponding to an IPv6 address as the URL specified for *new-path*.

When the request is received in the old path, the specified status code and the response with the new header path set in the Location header is returned. Normally, the Web browser that receives the status code 300, automatically redirects the request to the specified address in the Location header.

With the Redirect directive, you can redirect the request for a specific file to the specific file, or can specify to redirect the request for an optional path under a specific directory to the path with the same name under a different directory. Use the RedirectMatch directive when you want to redirect the request for an optional path under a specific directory to a specific file.

permanent: Responds with the status code 301 Moved Permanently.

temp: Responds with the status code 302 Found.

seeother: Responds with the status code 303 See Other.

gone: Responds with the status code 410 Gone. You cannot specify the new path.

status code: Responds with the specified status code. For details on values that you can specify, see *Appendix A Status codes*. However, when the status code other than 300 is specified, you cannot specify the new path.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
Redirect temp /index.html http://host number:port number/
default.html
```

Redirect the request for /index.html to the "http://host name:port number/default.html" with the status code 302.

(28) RedirectMatch [{permanent | temp | seeother | gone | status code}] regular-expressions new-path

(a) Contents

Specify this directive when the request from the client to the path that satisfies the condition mentioned in the regular expressions is to be requested again (redirect) for the new path.

Specify the old path of the request URL that starts with a slash in the regular expressions. Note that in old path, you cannot specify characters after the ? sign (query string).

You cannot specify a regular expression that is a duplicate of any of the following directive specification values:

- ProxyPass path name
- JkMount URL patterns in the redirector definition file

For example, the following regular expressions cannot be specified:

```
RedirectMatch ^/aaa/bbb/(.*) http://aaa.example.com/$1
ProxyPass /aaa/ http://aaa.example.com/
```

In the new path, specify the URL path that includes "protocol name://host name[:port number]". You can also specify an IPv6 address or the host name corresponding to an IPv6 address as the URL specified for *new-path*.

When the regular expressions are grouped using brackets ( ) , you can refer to the character string that matches with the group number i, using $i in the new path. Set the digits 1 to 9 for i. When a request to the path that satisfies the condition described in the regular expressions is received, the Web server responds with the specified status code and the Location header with new path set. Normally, the Web browser that receives the status code 300 automatically sends (redirects) the request to the address specified in the Location header.

For details on specifications of each status code, see *Redirect directive*.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

(Example 1)

```
RedirectMatch ^/other/ http://www.example.com/
```

Redirect all the requests that start with /other/ to "http://www.example.com/" with the status code 302.

(Example 2)

```
RedirectMatch permanent ^/old/(.*) http://www.example.com/
new/$1
```

Redirect the request for "/old/file name" to "http://www.example.com/new/file name" with the status code 301.

(29) RequestHeader {{set | append | add} header header-value [env=[!] environment-variable] | unset header}

(a) Contents

Specify this directive when you customize the header value received from client.

set: Sets the header. When header exists, rewrite it with the specified header value.

append: Adds the header value to the existing header. A comma delimits the existing header values. Set the header if it does not exist.

add: Sets a header in another line if the header already exists. Use to specify the same header in multiple lines.

unset: If the specified header exists, deletes it.

env=*environment-variable*: Executes the contents specified in the RequestHeader directive when the specified environment variable is set.

env=`!`*environment-variable*: Executes the contents specified in the RequestHeader directive when the specified environment variable is not set.

If *header-value* contains spaces, you must enclose the value in double quotation marks ("). For *header-value*, you can specify a character string containing only characters, a character string containing format identifiers, or a character string containing both characters and format identifiers. The format identifiers are as follows:

Format identifier	Description
%t	Displays the request reception time as the time elapsed from January 1, 1970, 00:00:00 GMT (Greenwich Mean Time). The unit is microseconds. `t=` is added at the beginning.

Format identifier	Description
%D	Displays the time taken for request processing. The unit is microseconds. `D=` is added at the beginning.
%{env_name}e	The value of the env_name environment variable

(b) Note

The mod_headers module must be embedded to use header customization functionality. For details on header customization functionality, see *4.10 Header customization functionality*.

(c) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(d) Overwrite permission

FileInfo level

(e) Specification example

```
RequestHeader set Host www.example.com
```

(30) Require {user user-name [user-name ...] | group group-name [group-name ...] | valid-user | file-owner | file-group}

(a) Contents

Specify this directive along with the AuthName directive, AuthType directive, AuthUserFile directive (or AuthGroupFile directive). This directive defines the access control.

user: Among the users registered in the password file specified with the AuthUserFile directive, only the users that are specified in the user names can access.

group: Only the users belonging to the group specified in the registered group name in the group file specified with the AuthGroupFile directive can access.

valid-user: All the users registered in the password file specified with the AuthUserFile directive, or the users of the group specified with the LDAPRequire directive can access. You cannot combine the password file and the LDAPRequire directive. Operations are not guaranteed when the two are combined.

file-owner: Among the users registered in the password file specified with the AuthUserFile directive, only the users that match with the owner of the system of the access target file, can access (can not specify in Windows version).

file-group: Among the users belonging to the group specified in the group names registered in the group file that is specified with the AuthGroupFile directive, only those users having a group name matching with the owner of the system of the access target file, can access (cannot specify in Windows version).

(b) Location where you can code

<Directory>, .htaccess

(c) Overwrite permission

AuthConfig level

6.2.7 Directives starting with S

(1) Satisfy {any | all}

(a) Contents

When the access to the contents is controlled by both the user authentication (specifications of AuthUserFile and Require directives) and the host name or the IP address (specifications of Allow from and Deny from directive), this directive sets their relationship.

any: If any of the condition is satisfied, permits the access to the contents.

all: If none of the conditions are satisfied, prohibits the access to the contents.

(b) Location where you can code

<Directory>, .htaccess

(2) Script Method CGI-script-name

(a) Contents

When a request based on the specified method, this directive executes the script displayed in CGI script name.

Methods that you can specify: GET, POST, PUT, DELETE

The Method is case sensitive.

However, in the case of GET method, the script is called only when there is an query string (for example, /foo.html?bar).

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>

(c) Specification example

```
Script POST /cgi-bin/search
```

(3) ScriptAlias URL directory-name

(a) Contents

This directive specifies the name of the directory that contains the CGI program to be executed for the requests to execute the CGI programs specified in the URL from the

Web browser.

You cannot specify a URL that is a duplicate of any of the following directive specification values:

- ProxyPass path name
- JkMount URL patterns in the redirector definition file

For example, the following URLs cannot be specified:

```
  ScriptAlias /aaa/bbb/ C:/alias/

 ProxyPass /aaa/ http://aaa.example.com/
```

Specify the directory name with an absolute path.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
ScriptAlias /cgi-bin/ "<Cosminexus-installation-directory>/
httpsd/cgi-bin/"
```

(4) ScriptAliasMatch regular-expression new-path

(a) Contents

When the URL requesting the execution of the specified CGI program from the Web browser satisfies the conditions described by the regular expressions, this directive executes the CGI program with the specified new path. When the regular expressions are grouped using brackets ( ), you can refer to the character string that matches with the expression of group i using $i in new path. Specify numeric characters from 1 to 9 for i.

Specify the new path with an absolute path. When '$' or '&' are included as characters of new path, add '￥' before the characters. Note that when you specify $i, you need not add '￥' before '$'.

You cannot specify a regular expression that is a duplicate of any of the following direcrive specification values:

- ProxyPass path name
- JkMount URL patterns in the redirector definition file

For example, the following regular expressions cannot be specified:

```
  ScriptAliasMatch ^/aaa/bbb/(.*) C:/alias/$1
  ProxyPass /aaa/ http://aaa.example.com/
```

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
ScriptAliasMatch ^/cgi-bin/(.*)
"<Cosminexus-installation-directory>/httpsd/cgi-bin/$1"
```

(5) ScriptInterpreterSource { registry | <u>script</u> } W

(a) Contents

This directive defines the interpreter used to execute the CGI script.

registry: The registry is searched and the program related to the extension is used as the interpreter.

script: The interpreter specified in #! Line in the script is used.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(6) ScriptLog file-name

(a) Contents

The ScriptLog directive specifies the file to which the CGI script error log is output. For the file name, you can specify either an absolute path, or the relative path from the value specified in the ServerRoot directive.

In UNIX, the specified file must be writable with the user authentication specified in the User directive.

(b) Location where you can code

httpsd.conf

(7) ScriptLogBuffer number-of-buffers

~((0 - 2147483647))<<1024>> (Unit: Bytes)

(a) Contents

This directive specifies the maximum value when collecting the log of request body by PUT and POST methods, in bytes. This specification in valid only when you specify the file of the error log output destination with the ScriptLog directive.

The area of the value specified in this directive is stored in the request process. As a result, when you specify large values, memory storage failure may occur and Web server may stop. Hitachi recommends that you specify the default value or the minimum required value.

(b) Location where you can code

httpsd.conf

(8) ScriptLogLength file-size

~((0 - 2147483647))<<10385760>> (Unit: Bytes)

(a) Contents

This directive specifies the maximum size of error log file of CGI script in bytes. The specification of this directive is valid only when you specify the error log output destination file with the ScriptLog directive.

(b) Location where you can code

httpsd.conf

(9) ServerAdmin E-Mail-address

(a) Contents

This directive specifies the E-Mail address of the server administrator. Always specify this directive when you specify the E-Mail address with the ServerSignature directive.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
ServerAdmin www-admin@server.example.com
```

(10) ServerAlias host-name [host-name ...]

(a) Contents

This directive specifies an optional name for the host name (ServerName) that is used in the virtual host based on server name. You can specify the host name corresponding to an IPv6 address.

(b) Location where you can code

<VirtualHost>

(11) ServerName server-name[:port-number]

(a) Contents

The ServerName directive specifies the server name and the port number of Cosminexus HTTP Server. When you omit the port number, the value specified in the Port directive is set.

Server name is specified in FQDN (fully qualified domain name) or IP address. You can also specify an IPv6 address or the FQDN corresponding to an IPv6 address for *server-name*. When specifying both an IPv6 address and a port number, enclose the

IPv6 address in square brackets ([ ]).

Based on the specified value of the UseCanonicalName directive, any requests that use an image map or any specifications of a directory that does not end with a / (slash) are set in the Location header as redirect destination (when redirect is indicated in the Web server) and returned to the client. Therefore, you must specify the server name that a client can access. Specification of this directive is mandatory.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
ServerName www.example.com
ServerName 2001::123:4567:89ab:cdef
ServerName [2001::123:4567:89ab:cdef]
ServerName [2001::123:4567:89ab:cdef]:8080
```

(12) ServerPath path-name

(a) Contents

Specify this directive when you use the path name instead of Host header to connect to each host, in the virtual host based on the server name.

(b) Location where you can code

<VirtualHost>

(13) ServerRoot directory-name

~<</opt/hitachi/httpsd>>(UNIX version)

~<<<*Cosminexus-installation-directory*>\httpsd>> (Windows version)

(a) Contents

This directive specifies the root directory of server with the absolute path.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
ServerRoot "C:/Program Files/Hitachi/Cosminexus/httpsd"
```

(14) ServerSignature {On | <u>Off</u> | Email}

(a) Contents

This directive specifies whether a content footer of error messages that the Web server creates is to be signed.

On: Displays the character string according to ServerTokens directive (such as

Cosminexus HTTP Server and version number), and the server name and the port name according to the specified value of UseCanonicalName directive.
```
Cosminexus HTTP Server 09-00 at www.example.com Port 80
```

Off: Does not display signature in the contents footer.

Email: In addition to the display when `On` is specified, this option adds the specification value of the ServerAdmin directive in a mailto tag.

Note that when `On` is specified, the IPv6 address specified in the ServerName directive or the host name corresponding to an IPv6 address can be displayed.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Specification example
```
ServerSignature On
```

(15) ServerTokens {Minimal | *OS* | *Full* | *ProductOnly}*

(a) Contents

This directive sets the server header format of the HTTP response header. The Server header values based on respective setting are described below. Unix, Win32, or Win64 is set as the OS type. The value of Server header is used as per the client specifications.

Minimal: Cosminexus HTTP Server Version number

OS: Cosminexus HTTP Server Version number (OS type)

Full: Cosminexus HTTP Server Version number (OS type) Information set by an additional PP

ProductOnly: Cosminexus HTTP Server

(b) Location where you can code

httpsd.conf

(c) Specification example
```
ServerTokens Full
```

(16) SetEnv environment-variable-value

(a) Contents

This directive specifies the environment variable value that is set when you pass the optional environment variable to the CGI script. If you specify this directive multiple times, you cannot specify different values for the same environment variable.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
SetEnv MY_ENV myenv
```

(17) SetEnvIf request-value regular-expression environment-variable[=value] [environment-variable[=value] ...]

(a) Contents

This directive defines the environment variable on the basis of the client request. Set the specified environment variable when the request value from client satisfies the conditions in the regular expressions. By default, the value is set to `1`. When ! is added before the environment variable, the setting of that environment variable is cancelled. You can specify the value shown in the HTTP request header or the values shown in the following table as the request value. You can search an environment variable by specifying the environment value specified earlier as the request value. However, such environment variable must not conform to the HTTP request header and to the values described in the following table:

Request value	Meaning
Remote_Addr	IP address of the client
Remote_Host	Host name of the client (only when set in the request)
Request_Protocol	Protocol of the request (such as HTTP/1.1)
Request_Method	Method name of the request (such as GET, POST, and HEAD)
Request_URI	URI of the request
Server_Addr	IP address of the server that receives the request

When performing multiple specifications of this directive, you cannot specify the same request value multiple times.

Note that when specifying Remote_Host for *request-value*, you can also specify the host name corresponding to an IPv6 address for *regular-expression*. In addition, the Remote_Addr and Server_Addr request values cannot be used for connections that use IPv6. To use Remote_Addr and Server_Addr, perform settings by using the HWSSetEnvIfIPv6 directive.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

(Example 1)

```
SetEnvIf User-Agent "Mozilla.*" SETENVIF_USER_AGENT=Mozilla
```

(Example 2)

```
SetEnvIf Request_URI "¥.(gif)|(jpg)$" request_is_image
```

(Example 3)

For connections that use IPv4, set the environment variable for a specific client as follows:

```
Listen 123.123.123.123:80
Listen [2001::123:4567:89ab:cdef]:80
<VirtualHost 123.123.123.123:80>
        SetEnvIf Remote_Addr ^234¥.234¥.234¥.234$
IPV4_CLIENT
</VirtualHost>
```

(18) SetEnvIfNoCase *request-value regular-expression environment-variable[=value] [environment-variable[=value] ...]*

(a) Contents

This directive defines the environment variables based on the client request. Set the specified environment variables when the request value from the client satisfies the conditions described in the regular expression. By default, the value is set to `1`. When ! is attached before environment variable, it cancels the settings of that environment variable.

For details on the values that can be specified for *request-value*, see the SetEnvIf directive.

However, *regular-expression* is not case-sensitive in this directive. When specifying this directive multiple times, you cannot specify the same request value in the directives.

Note that when specifying Remote_Host for *request-value*, you can specify the host name corresponding to *request-value*. In addition, the Remote_Addr and Server_Addr request values cannot be used for connections that use IPv6. To use Remote_Addr and Server_Addr, perform settings by using the HWSSetEnvIfIPv6 directive.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(19) SetHandler handler-name

(a) Contents

Specify this directive when the requests of specified <Directory> or all the requests in the scope of access control file are related to the handlers specified with the handler names. If you specify `none` as the handler name, the settings specified for the SetHandler directive until then are disabled.

(b) Location where you can code

<Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(20) SSLBanCipher encryption-type [encryption-type ...]

(a) Contents

This directive denies an access to specified encryption types and returns the status code 403 Forbidden to the client. The following are the encryption types.

Encryption type	Key exchange method	Authentication method	Symmetric key cryptography	Encryption key size (bit)	Message authentication algorithm
RC4-MD5	RSA	RSA	RC4	128	MD5
RC4-SHA	RSA	RSA	RC4	128	SHA
AES128-SHA	RSA	RSA	AES	128	SHA
AES128-SHA256	RSA	RSA	AES	128	SHA256
DES-CBC3-SHA	RSA	RSA	DES	168	SHA
AES256-SHA	RSA	RSA	AES	256	SHA
AES256-SHA256	RSA	RSA	AES	256	SHA256

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(21) SSLCACertificateFile file-name

(a) Contents

When you perform the server and client authentication by SSL, this directive specifies the file name of the public key (PEM format) of the CA (Certification Authority). By combining multiple certificate files, you can mix multiple certificates into one file.

Specify the file name with an absolute path.

- Usage during server authentication

  When operating with the server certificate issued by the chained CA, this directive sets the certificate of chained CA.

- Usage during client authentication

  This directive sets the certificate of CA that issues the client certificate. It also sets the certificate of chain CA for the client certificate that is chained.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
SSLCACertificateFile "<Cosminexus-installation-directory>/
httpsd/conf/ssl/cacert/anycert.pem"
```

(22) SSLCACertificatePath directory U

(a) Contents

When performing the server and client authentication by SSL, this directive specifies the directory that stores the hash link of the CA certificate (PEM format). For details on how to create and operate a hash link, see *5.2.6 Creating a hash link (in UNIX)* (certutil cert command).

You need the hash link when verifying the client certificate. For the acquired CA certificate, create a hash link in a specific directory and specify this directory in the SSLCACertificatePath directive.

Specify directory name with an absolute path.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
SSLCACertificatePath /opt/hitachi/httpsd/conf/ssl/cacerts
```

(23) SSLCacheServerPath path-name

(a) Contents

This directive specifies the path name for the SSL session management cache server gcache. Specify the path name with an absolute path, or the relative path from the ServerRoot directive.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
SSLCacheServerPath /opt/hitachi/httpsd/sbin/gcache
```

(24) SSLCacheServerPort {port-number **|** *path-name}*

~((Use 1 - 65535 when specifying the port number))

(a) Contents

This directive specifies the port number or the path name to exchange data between Web server and SSL session management cache server gcache. You can specify the path name with an absolute path, or the relative path from the ServerRoot directive.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
SSLCacheServerPort logs/gcache_port
```

(25) SSLCacheServerRunDir path-name

~<<Specification value of the ServerRoot directive>>

(a) Contents

This directive specifies the path name to run the SSL session management cache server gcache. Use gcache to specify the directory to output the core dump. You can specify path name with an absolute path, or the relative path from the ServerRoot directive. You need to grant the read permissions, write permissions, and execution permissions specified in the User directive to the directory specified in the path name.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
SSLCacheServerRunDir /opt/hitachi/httpsd/logs
```

(26) SSLCertificateFile file-name

(a) Contents

This directive specifies the file name of the Web server certificate (PEM format) when you perform the authentication by SSL.

Specify file name with an absolute path.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
SSLCertificateFile "<Cosminexus-installation-directory>/
httpsd/conf/ssl/server/httpsd.pem"
```

(27) SSLCertificateKeyFile file-name

~<<Specification value of the SSLCertificateFile directive>>

(a) Contents

This directive specifies the private key file name of the Web server when you perform the authentication by SSL.

Specify the file name with an absolute path.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
SSLCertificateKeyFile "<Cosminexus-installation-directory>/
httpsd/conf/ssl/server/httpsdkey.pem"
```

(28) SSLCertificateKeyPassword path-name

(a) Contents

This directive specifies the path name of the file that stores the password of the server private key where the password is protected. The File that stores the password is created by the `sslpasswd` command. Specify the path name with an absolute path, or the relative path from the ServerRoot directive.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(29) SSLCRLAuthoritative {On | Off}

(a) Contents

Specify this directive to handle the next issue date of the CRL that is used during the SSL client authentication.

On:

During the SSL client authentication, when the next issue date of the CRL corresponding to client certificate elapses, the Web server denies the connection to the client due to a failure in client authentication. Correct operation of the CRL is required.

Off:

Ignores the next issue date of the CRL. If the next issue date elapses but CRL is valid, the client can connect even though not registered in CRL. Security level will go down, however, even if the CRL is not operated properly, you can continue the service by maintaining the minimum level of security.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
SSLCRLAuthoritative On
```

When the next issue date of the CRL elapses, deny the access to the clients holding a certificate issued by the CA that also issues the CRL.

(30) SSLCRLDERPath path-name

(a) Contents

This directive specifies the directory that stores the CRL of the DER format with an absolute path. If you store the required CRL in the specified directory and start or restart the Web server, you can apply the CRL during the client authentication with SSL. When the CRL issued from the same CA (including the CRL in the directory specified with the SSLCRLPEMPath directive) is stored multiple times, the CRL with latest date is applicable. The Web server does not start when directory contains a file other than the CRL of the DER format.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
SSLCRLDERPath "<Cosminexus-installation-directory>/httpsd/
conf/ssl/crl/DER"
```

Specify the directory in which the CRL file of the DER format is stored.

(31) SSLCRLPEMPath path-name

(a) Contents

This directive specifies the directory that stores the CRL of the PEM format with an absolute path. If you store the required CRL in the specified directory and start or restart the Web server, you can apply the CRL during the client authentication with SSL. When the CRL issued from the same CA (including the CRL in the directory specified with the SSLCRLDERPath directive) is stored multiple times, the CRL with latest date is applicable. The Web server does not start when directory contains a file other than the CRL of the PEM format.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
SSLCRLPEMPath "<Cosminexus-installation-directory>/httpsd/
conf/ssl/crl/PEM"
```

Specify the directory in which CRL file of the PEM format is stored.

(32) SSLDenySSL

(a) Contents

Specify this directive to prohibit the SSL access. When you specify this directive, and even if the SSL is enabled with the SSLEnable directive, access from https is denied with the status code 403 Forbidden. The operations contrary to the SSLRequireSSL directive take place.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <directory>, .htaccess

(c) Overwrite permission

FileInfo level

(33) SSLDisable

(a) Contents

This directive disables the SSL. The default value is the SSLEnable directive (SSL is enabled). Specify this directive to disable the SSL for a specific host with the virtual host.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(34) SSLEnable

(a) Contents

This directive enables the SSL. By default, the SSL is enabled unless the SSLDisable directive is specified.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(35) SSLExportCertChainDepth value

~((0 - 9))<<0>>

(a) Contents

In the case of performing the SSL client authentication, specify this directive when setting the certificate from the CA that issued the client certificates in the environment variable SSL_CLIENT_CERT_CHAIN_n, up to the root CA. The maximum specification value is n. This directive is enabled only when the SSLExportClientCertificates directive is specified. As the specified number of CA certificates are cached in the gcache server, you can use the cache effectively by specifying only the required number of CGI programs or Servlets in this directive. However, when some of the cached certificates are deleted due to memory restrictions and cannot be acquired, only those certificates that could be acquired are set in the environment variable.

0:

Environment variable is not set.

1~9:

Numbers are allocated from the CA close to the client in a sequence and the environment variable is set. In the environment variable, the Base64 encoded value of the certificate in the DER format is set. Number of bytes when a certificate is encoded in Base64, is around 1KB.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

This example describes a certificate chain called the '*Root CA-Lower CA-Client certificate*'. In this case, the relationship between the environment variables and the certificates is as follows:

Environment variable	Certificate
SSL_CLIENT_CERT	Client certificate
SSL_CLIENT_CERT_CHAIN_1	Lower CA certificate
SSL_CLIENT_CERT_CHAIN_2	Root CA certificate

Specify the directive in the following format to acquire all environment variables and certificate chains:

```
SSLExportClientCertificates
SSLExportCertChainDepth 2      Specify the value of 2 or more
```

(36) SSLExportClientCertificates

(a) Contents

In the case of performing the SSL client authentication, specify this directive when setting the client certificate in the environment variable SSL_CLIENT_CERT. In the environment variable SSL_CLIENT_CERT, set the Base64 encoded value of the certificate in DER format

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(37) SSLFakeBasicAuth

(a) Contents

Along with the SSL client authentication functionality, specify the settings such that the Basic authentication is possible only by presenting the client certificate without entering user ID and password in the Web browser. Code the Subject and the password of X509 client certificate in the file specified with the AuthUserFile directive. The password is always fixed to the following values ("password" is encrypted in both):

- UNIX Version: "xxj31ZMTZzkVA"
- Windows Version: "{SHA}W6ph5Mm5Pz8GgiULbPgzG37mj9g="

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

The following shows the value of the Subject field in the certificate that is displayed by the `certutil` command.

```
Subject:
EMAIL=username@userhost,CN=username,OU=Software,O=HITACHI,L
=Yokohama-shi,ST=Kanagawa,C=JP
```

In this case, the file specified with the AuthUserFile directive is specified as follows:

In UNIX Version

```
/C=JP/ST=Kanagawa/L=Yokohama-shi/O=HITACHI/OU=Software/
CN=username/EMAIL=username@userhost:xxj31ZMTZzkVA
```

In Windows Version

```
/C=JP/ST=Kanagawa/L=Yokohama-shi/O=HITACHI/OU=Software/
CN=username/
EMAIL=username@userhost:{SHA}W6ph5Mm5Pz8GgiULbPgzG37mj9g=
```

Subject is logged in the u specification of the LogFormat directive.

If the authentication fails, the Web server responds with the status code 401 Authorization Required.

(38) SSLProtocol protocol-name [protocol-name ...]

~<<All>>

(a) Contents

This directive specifies the version of SSL protocol that is used.

You can specify the following values as protocol name:

SSLv3: Use the SSL protocol version 3.

TLSv1: Use the TLS protocol version 1.0.

TLSv11: Use the TLS protocol version 1.1.

TLSv12: Use the TLS protocol version 1.2.

All: Use all the above-mentioned protocols.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(39) SSLRequireCipher encryption-type [encryption-type ...]

(a) Contents

This directive denies the access to encryption types other than the specified ones, and returns the status code 403 Forbidden to the client. For the encryption code that you can specify, see *SSLBanCipher directive*.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite Permission

FileInfo level

(40) SSLRequiredCiphers encryption-type [:encryption-type ...]

(a) Contents

This directive specifies the encryption types that you can use in the SSL communication. If the encryption types specified in this directive and the encryption types that the client can use match, the Web server establishes the SSL communication and receives the HTTP requests. If encryption types do not match, the Web server does not establish the SSL communication or receive the HTTP request. For details on the encryption types that you can specify, see *SSLBanCipher directive*.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
SSLRequiredCiphers
RC4-MD5:RC4-SHA:DES-CBC3-SHA:AES128-SHA:AES256-SHA
```

(41) SSLRequireSSL

(a) Contents

Specify this directive when allowing the access only through SSL. When this directive is specified, even if the SSL is disabled with the SSLDisable directive, access from http is denied with the status code 403 Forbidden. This directive prevents the exposure of the contents due to carelessly disabled SSL in the coding locations of different directives.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
<VirtualHost 172.17.40.10:443>
 SSLDisable
 ...
  <Directory /secure/dir>
    SSLRequireSSL
    ...
  </Directory>
</VirtualHost>
```

This example allows the http access to the port 443 of 172.17.40.10 host but denies the access to directory /secure/dir. The Web server responds with the status code 403 Forbidden for the http access to /secure/dir directory.

(42) SSLSessionCacheTimeout value

~((1 - 2147483647))<<3600>> (Unit: Second)

(a) Contents

This directive sets the validity period (in seconds) for the data such as the session ID that is maintained by the Web server, or in SSL session control cache server gcache.

(b) Note

Set the validity period in such a way that it does not exceed January 19, 2038, 03:14:07 of the Greenwich Mean Time (GMT).

(c) Location where you can code

httpsd.conf, <VirtualHost>

(d) Specification example

```
SSLSessionCacheTimeout 3600
```

(43) SSLSessionCacheSize {size | max}

~((0 - 2147483647))<<16777216>> (Unit: Bytes)

(a) Contents

In UNIX version, this directive sets the upper limit for the memory size in bytes, for the data such as session IDs cached in the memory of the cache server gcache that manages the SSL session. When you set this directive to `0`, gcache server does not start and hence the session cache does not operate.

In Windows version, this directive specifies the upper limit of session cache size in bytes. When `0` is specified, session cache in not implemented.

When `max` is specified, the upper limit of session cache size is not set. For each SSL session, approximately 200 bytes are used when performing only the server authentication, and approximately one kilobyte is used when performing the client authentication also.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
SSLSessionCacheSize 1024
```

(44) SSLSessionCacheSizePerChild {size | max} U

~((0 - 2147483647))<<20480>> (Unit: Bytes)

(a) Contents

This directive specifies the upper limit of the memory size for the data such as the session IDs that are cached in the server process memory. When you specify `max`, the upper limit is not set.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
SSLSessionCacheSizePerChild 1024
```

(45) SSLVerifyClient {0 | 1 | 2}

~<<0>>

(a) Contents

This directive specifies the settings for certificate used during the client authentication.

0: Certificate is not requested.

1: Client can display the certificate. This setting is used to test the operations.

2: Client must display the certificate.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
SSLVerifyClient 2
```

(46) SSLVerifyDepth number-of-levels

~((0 - 10))<<0>>

(a) Contents

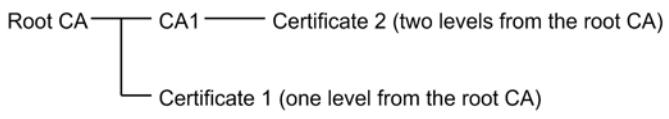
This directive specifies the number of levels up to which the certificate chain is traced.

Specify the number of levels for authentication check related to CA certificate chain used in client authentication. This directive is used to limit the extent up to which the CA chain is used. Specify minimum two levels since the certificates with self-signatures are not authenticated. For example:

(Example)

Conditions

- CA1 has signed the root CA.
- Certificate 1 signs in root CA.
- Certificate 2 signs in CA1.


Root CA
CA1
Certificate 2 (two levels from the root CA)
Certificate 1 (one level from the root CA)

Specifying SSLVerifyDepth

In the above example, specify minimum three levels in the SSLVerifyDepth directive to execute the authentication check for both certificates 1 and 2. Specify 2 in SSLVerifyDepth directive, to enable the authentication check for certificate 1 but not for certificate 2.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
SSLVerifyDepth 10
```

(47) StartServers number-of-processes Ⓤ

~((0 - 1024))<<5>>

(a) Contents

This directive specifies the number of server processes when you start the Web server. For details on the other directives related to the number of processes, see *4.1 Relationship between processes and directives of Cosminexus HTTP Server*.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
StartServers 5
```

6.2.8 Directives starting with T and U

(1) Timeout time

~((0 - 65535))<<300>> (Unit: Seconds)

(a) Contents

The Timeout directive specifies the following waiting times in seconds. If `0` is specified, the waiting time is 0 seconds.

- When the Web server is receiving requests from the client, this directive specifies the waiting time for receiving the request when the data is not received (receiving the HTTP protocol after establishing the connection).
- When the Web server is sending a response to the client, this directive specifies the waiting time for sending the response when the data cannot be sent.
- When the Web server is sending a request to a CGI program, this directive specifies the waiting time for sending the request when the data cannot be sent.
- The waiting time for receiving a response after sending the request to a CGI program.
- When the Web server is receiving the response from CGI program, this directive specifies the waiting time to receive the request when the data is not received.
- The waiting time after receiving a response from the CGI program until the I/O pipe is closed
- When using a reverse proxy, the waiting time after data could not be transmitted while sending a request to the backend server

- When using a reverse proxy, the waiting time until a response is received after a request is sent to the backend server
- When using a reverse proxy, the waiting time after data could not be received while receiving a request from the backend server

(b) Location where you can code

httpsd.conf

(c) Specification example

```
Timeout 300
```

(2) ThreadsPerChild number-of-threads

~((1 - 1024))<<40>>

(a) Contents

This directive specifies the number of threads to be started as a server. The number of specified threads denotes the maximum number of concurrent server connections.

(b) Location where you can code

httpsd.conf

(3) TraceEnable {On | Off | extended}

(a) Contents

This directive specifies whether to deny the requests from the TRACE method.

On: Grants permission to the request from the TRACE method. However, when the request body is added, the Web server responds with the status code 413 Request Entity Too Large.

Off: Denies the request from the TRACE method. When request is from the TRACE method, the Web server responds with the status code 403 Forbidden.

Extended: Grants permission to requests from the TRACE method. Grants permission even if the request body has been added. However, the upper limit of the size of request bodies other than request bodies from a reverse proxy is 64 KB.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
TraceEnable Off
```

(4) TransferLog {file-name | pipe}

(a) Contents

This directive specifies a file in which the log is stored or a program that outputs log. You can specify the log format with the LogFormat directive that does not specify a label name.

If you specify the log format in the LogFormat directive, an IPv6 address and the host name corresponding to the IPv6 address can be output. For details on the formats that you can specify, see the CustomLog directive.

If the log format is not specified in the LogFormat directive, the log is output in a standard log format.

file-name: Specifies a file name that stores the log. In file name, you can specify the absolute path or the relative path from the specified value of the ServerRoot directive.

pipe: Specifies a program that receives the log information from the standard input in "| Program name" format. For more details regarding the notes on Windows version, see *CustomLog directive*.

(b) Location where you can code

httpsd.conf, <VirtualHost>

(c) Specification example

```
TransferLog "|¥"¥"<Cosminexus-installation-directory>/httpsd/
sbin/rotatelogs.exe¥" ¥ "<Cosminexus-installation-directory>/
httpsd/logs/access¥" 86400¥""
```

Use the rotatelogs program to divide and collect the log every 24 hours.

(5) TypesConfig file-name

~<<conf/mime.types>>

(a) Contents

This directive specifies the settings file that defines the relationship between the file extension and contents type (MIME type). In file name, you can specify the absolute path or the relative path from the specified value of the ServerRoot directive.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
TypesConfig conf/mime.types
```

The settings file for MIME types is mime.types

(6) UnsetEnv environment-variable [environment-variable ...]

(a) Contents

The UnsetEnv directive specifies the environment variables to delete from the environment variables to be passed to CGI scripts that are specified in the SetEnv or PassEnv directive.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>, .htaccess

(c) Overwrite permission

FileInfo level

(d) Specification example

```
UnsetEnv MY_ENV
```

(7) UseCanonicalName {On | Off | dns}

(a) Contents

This directive specifies the generation method of the standard name of the server. The standard name of the server is set in the URL and the SERVER_NAME and SERVER_PORT of the environment variable that refer to local server.

On: The standard name of server is created from the ServerName directive value and is set in the URL and environment variable that refer to the local server. When you use the IP address to specify the VirtualHost, specify the ServerName in the VirtualHost block. When the ServerName is not specified in the block, acquire the host name from the IP address.

Off: The standard name of the server is created from the host name and the port number that is provided from the client by the Host header, and is set in the URL and environment variable that refers to the local server. However, when Host header is not provided, the standard name is created using the ServerName directive and the port name that is used in the actual connection.

dns: This option is for the old client that does not have the Host header. When specifying this option, the standard name of the sever is created using the host name extracted from the server ip address received from the client and the port number used in the actual connection, and set in the URL and environment variable that refers to the local server.

Note that IPv6 addresses are supported when the `On`, `Off`, or `dns` option is selected.

(b) Location where you can code

httpsd.conf, <VirtualHost>, <Directory>

(8) User user-name U

~<<#-1>>

(a) Contents

This directive specifies the user name when server process is running.

(b) Location where you can code

httpsd.conf

(c) Specification example

```
User nobody
```

(9) UserDir {directory-name **|** *disabled [user-name [user-name ...]]}*

~<<public_html>>(UNIX Version)

~<<disabled>>(Windows Version)

(a) Contents

This directive specifies a location on the server that is released for the request to /~user name/ from the Web browser as the directory name. When you set this directive as disabled, you can specify the user to whom the Web contents are not disclosed.

Specify the directory name using the relative path or the absolute path.

In Windows version, only the absolute path is valid.

directory-name:

- When specified by the relative path

  This option specifies a location when the users with the user IDs on the server release the Web contents under the user home directory. When there is a request for the /~user name/, the Web server accesses the "home directory of user/ directory name".

- When specified by absolute path

  This option specifies a location for the user directory. When there is a request for /~user name/, the Web server accesses the "directory name/user name".

disabled:

This option specifies a user that does not release Web contents for the request to the /~user name/ from Web browser. The directory name to be accessed is not changed for the request of the specified user name. When the user name is not specified, `disabled` is set for all users.

(b) Note

- When you specify the directory name using multiple UserDir directives, the directory name specified later overwrites the earlier specifications.
- You can specify the user names to be specified in `disabled` using multiple UserDir directives.

(c) Location where you can code

httpsd.conf, <VirtualHost>

(d) Specification example

(Example 1)

```
UserDir public_html
```

If home directory of user user1 is /home/user1, access /home/user1/public_html/index.html in the request http://host name [:port number]/~user1/index.html.

(Example 2)

```
UserDir /home
UserDir disabled user3
UserDir disabled user4 user5
```

Access /home/user1/index. html in the request http://host name [:port number]/~user1/index.html. However, you cannot access /home/user3/index.html in the requests http://host name [:port number]/~user3/index.html, since user3 is disabled. The access regarding user4 and user5 are similar to user3.

(Example 3)

```
UserDir disabled
```

Specify `disabled` for all users.

Appendixes

A. Status codes

The following table describes the status codes that Cosminexus HTTP Server returns to the Web browser. The status code is returned by HTML charset=ISO-8859-1.

When Cosminexus HTTP Server returns a status code to the Web browser, an error message is also generated automatically according to the status code and is returned as HTML encoded in ISO-8859-1.

Table A-1: List of status codes

Status code	Contents
100 Continue	Client can continue requests.
200 OK	Normal Exit
204 No Content	The request has ended normally, but there is no resource to return. The Web server generates the status code as per specifications of the ImapDefault nocontent directive.
206 Partial Content	Returns the partial resource. The Web server generates the status code when partial contents are returned to the Partial GET request that uses the client Range header.
300 Multiple Choices	Multiple pages can be made available.
301 Moved Permanently	The resource is moved permanently. The Web server generates this status code based on the request http://*Host-name* [:*Port-number*]/*Directory-name* for the directory that is not closed by a slash, and as per specifications of the Redirect permanent directive.
302 Found	The resource is moved temporarily. The Web server generates the status code based on specifications of the Redirect temp directive.
303 See Other	The resource is moved. The Web server generates the status code based on specifications of the Redirect see other directive.
304 Not Modified	The requested contents are not changed.

Status code	Contents
400 Bad Request	The request has a syntax error. This status code is generated when: • A wrong header is specified. • There is no host header in HTTP/1.1. • The number of requested headers exceeds the value of the `LimitRequestFields` directive. • The `CONNECT` method was used to send a request to a static content file or a CGI program located on the Cosminexus HTTP Server.
401 Authorization Required	Authentication is required to access resources. This status code occurs when access is controlled by the AuthName directive or AuthUserFile directive.
403 Forbidden	Access to the resource is forbidden. The Web server generates this status code when access is denied by the access control or there is a request to execute the CGI program without execution permission.
404 Not Found	The resource is not found. The Web server generates this status code when there is a request for a file that is not on the server.
405 Method Not Allowed	The client uses a method that is not allowed. For static content files on the Cosminexus HTTP Server, `GET`, `HEAD`, `POST`, `OPTIONS`, and `TRACE` are usable methods. In CGI programs, usable methods depend on the implementation of the CGI program.
406 Not Acceptable	The client cannot respond as per the type specified in the Accept header.
408 Request Time-out	The request has timed out.
410 Gone	The client cannot use the resource permanently. The Web server generates this status code based on specifications of the Redirect gone directive.
411 Length Required	Client needs to specify the Content-Length header.
412 Precondition Failed	Conditions specified in the If-Unmodified-Since header or in the If-Matched header of the client do not match.
413 Request Entity Too Large	The request body size is very large and the server cannot process it. The Web server generates this status code when the length of the request body is larger than the length specified in the LimitRequestBody directive.
414 Request-URI Too Large	The request URI is very large and the server cannot process it. The Web server generates this status code when the length of the URI that includes the inquiry character string is larger than the length specified in the LimitRequestLine directive.

Status code	Contents
416 Requested Range Not Satisfiable	The specification range of the Range header exceeds the corresponding resource range. The Web server generates this status code when all the following conditions are fulfilled: • The request includes the Range header field. • The specified value of the field range does not overlap the current range of the selected resource. • The request does not include the field of the If-Range request header.
417 Expectation Failed	The extension of Expect request header field is not received.
500 Internal Server Error	An Error has occurred on the Web server. The Web server generates this status code when there is a problem in the CGI program and when an error occurs in the access control file (`.htaccess`). The detailed information is output in the error log.
501 Method Not Implemented	The request was for an unsupported method.
502 Bad Gateway	The proxy server has received an incorrect request.
503 Service Temporarily Unavailable	The server cannot process the current request due to overloading.
506 Variant Also Negotiates	An internal deployment error occurs in the server.

#

The status codes mentioned in this table and other status codes are output from the top level CGI programs integrated with Cosminexus HTTP Server. In such cases, see the manuals of the respective programs.

When using the reverse proxy, status codes 400 Bad Request, 403 Forbidden, and 502 Bad Gateway may become status codes 400 Proxy Error, 403 Proxy Error, and 502 Proxy Error.

B. Environment variables passed to CGI programs

Table B-1, Table B-2, and Table B-5 describe the list of environment variables that the Web server passes to the CGI programs. Table B-3 and Table B-4 describe examples of SSL_SERVER_element and SSL_SERVER_I_element. There may be cases when environment variables coded here are not set and the environment variables that are not coded here are set, depending upon the platform, client settings, request format, and directive settings of the Web server. Server names, domain names, and mail addresses mentioned in the table are false values.

Table B-1: List of environment variables

Environment variable name	Contents	Example
AUTH_TYPE	Authentication type in the user authentication	Basic
COMSPEC	Executable file of the command prompt	C:\WINNT\system32\cmd.exe
CONTENT_LENGTH	Number of data bytes, when the request from client is POST	20
CONTENT_TYPE	Contents type, when the request from client is POST	application/ x-www-form-urlencoded
DOCUMENT_ROOT	Specification value of the DocumentRoot directive	<*Cosminexus-install-directory*>/ httpsd/htdocs
GATEWAY_INTERFACE	CGI version	CGI/1.1
HTTP_ACCEPT	Value of the Accept header that the client displays	image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/ png, */*
HTTP_ACCEPT_CHARSET	Value of the Accept-Charset header that the client displays	Shift_JIS,*,utf-8
HTTP_ACCEPT_ENCODING	Value of the Accept-Encoding header that the client displays	gzip
HTTP_ACCEPT_LANGUAGE	Value of the Accept-Language header that the client displays	ja,fr, en,it
HTTP_CONNECTION	Value of Connection header that the client displays	Keep-Alive
HTTP_HOST	Value of the Host header that the client displays	www.hws.hitachi.co.jp:8080

Environment variable name	Contents	Example
HTTP_PRAGMA	Value of the Pragma header that the client displays	no-cache
HTTP_REFERER	Value of the Referer header that the client displays	http://www.hws.hitachi.co.jp:8080/test.html
HTTP_USER_AGENT	Value of the User-Agent header that the client displays	Mozilla/4.73 [ja] (WinNT; U)
PATH	PATH information on Web server	C:\WINNT\system32;C:\WINNT;C:\WINNT\System32\Wbem
PATH_INFO	The part after the CGI script in a URL	/dir1/file1
PATH_TRANSLATED	PATH_INFO value converted into the file system	<*Cosminexus-install-directory*>\httpsd\htdocs\dir1\file1
QUERY_STRING	Query String sent from the client	query1=a&query2=b
REMOTE_ADDR	Client address	172.17.xx.xx
REMOTE_HOST	Host name of the client (except when the HostnameLookups is `Off` and the host name is resolved)	Hostxxx
REMOTE_IDENT	Client ID	Unknown
REMOTE_PORT	Port number of the client	2298
REMOTE_USER	Authenticated user name when the request is authenticated	Userxxx
REQUEST_METHOD	HTTP method sent by the client	GET
REQUEST_URI	Request URI sent by the client	/cgi-bin/test-cgi?query1=a&query2=b
SCRIPT_FILENAME	File name of the requested CGI script	<*Cosminexus-install-directory*>/httpsd/cgi-bin/test-cgi
SCRIPT_NAME	The URI of the requested CGI script	/cgi-bin/test-cgi
SERVER_ADDR	The IP address of the Web server	172.17.xx.xx
SERVER_ADMIN	Specified value of the ServerAdmin directive	www-admin@server.example.com
SERVER_NAME	Host name of the Web server	

SERVER_PORT	The port name of the Web server	8080
SERVER_PROTOCOL	HTTP version that the client displays	HTTP/1.0
SERVER_SIGNATURE	Signature of the Web server (including the HTML tag)	<ADDRESS>Cosminexus HTTP Server 09-00 at www.example.com Port 8080</ADDRESS>
SERVER_SOFTWARE	Program name of the Web server	Cosminexus HTTP Server 09-00
SYSTEMROOT	System directory	C:\WINNT
TZ	Time zone of the Web server	JST-9
WINDIR	System directory	C:\WINNT

Table B-2: List of environment variables for SSL communication

Environment variable name	Contents	Example
HTTPS	Displays the secure communication.	On
HTTPS_CIPHER	SSL encryption type	RC4-MD5
HTTPS_KEYSIZE	Number of bits of the key for the symmetric key cryptography	128
HTTPS_SECRETKEYSIZE	Number of valid bits amongst the bits of the key for the symmetric key cryptography	128
SSL_CIPHER	SSL encryption type (similar to the HTTPS_CIPHER)	RC4-MD5
SSL_PROTOCOL_VERSION	SSL protocol version	SSLv3
SSL_SERVER_DN	Distinguish Name of the subject of the SSL server certificate	/C=JP/ST=Kanagawa/ L=Yokohama-shi/O=HITACHI/ OU=WebSite/ CN=www.hws.hitachi.co.jp/ EMAIL=www-admin@hws.hitachi.co.jp
SSL_SERVER_*ELEMENT*	Each element of the Distinguish Name of the SSL server certificate subject	Table B-3 shows an example of when SSL_SERVER_DN is as shown above.
SSL_SERVER_I_DN	Distinguish Name of the SSL server certificate issuer	/C=JP/ST=Kanagawa/ L=Yokohama-shi/O=LOCAL-CA/ OU=ca1/CN=ca1.hitachi.co.jp/ EMAIL=ca-admin@ca1.hitachi.co.jp

Environment variable name	Contents	Example
SSL_SERVER_I_*ELEMENT*	Each request of the Distinguish Name of the SSL server certificate issuer	Table B-4 shows an example of when SSL_SERVER_I_DN is as shown above.
SSL_SESSION_ID	SSL session ID (hexadecimal)	F968F8D7075B76587F35931DC594D3E3
SSL_MES_VERSION	MES version	MES 3.2.2.2 27-July-2010

Table B-3: Examples of SSL_SERVER_ELEMENT

Environment variable name	Contents	Example
SSL_SERVER_C	Country Name of the subject of the SSL server certificate (Web server)	JP
SSL_SERVER_CN	Common Name of the SSL server certificate subject	www.hws.hitachi.co.jp
SSL_SERVER_EMAIL	E-Mail address of the SSL server certificate subject	www-admin@hws.hitachi.co.jp
SSL_SERVER_L	Locality Name of the SSL server certificate subject	Yokohama-shi
SSL_SERVER_O	Organization Name of the SSL server certificate subject	HITACHI,Ltd.
SSL_SERVER_OU	Organization Unit Name of the SSL server certificate subject	WebSite
SSL_SERVER_ST	State Name of the SSL server certificate subject	Kanagawa

Table B-4: Examples of SSL_SERVER_I_ELEMENT

Environment variable name	Contents	Value
SSL_SERVER_I_C	Country Name of the SSL server certificate issuer	JP
SSL_SERVER_I_CN	Common Name of the SSL server certificate issuer	ca1.hitachi.co.jp
SSL_SERVER_I_EMAIL	E-Mail address of the SSL server certificate issuer	ca-admin@ca1.hitachi.co.jp
SSL_SERVER_I_L	Locality Name of the SSL server certificate issuer	Yokohama-shi

Environment variable name	Contents	Value
SSL_SERVER_I_O	Organization Name of the SSL server certificate issuer	LOCAL-CA
SSL_SERVER_I_OU	Organization Unit Name of the SSL server certificate issuer	ca1
SSL_SERVER_I_ST	State Name of the SSL server certificate issuer	Kanagawa

Table B-5: List of environment variables when authenticating the SSL client

Environment variable name	Contents	Example
SSL_CLIENT_CERT	SSL client certificate (DER-BASE64 format) The setting of the SSLExportClientCertificates directive is required.	"MIIDrTCCAxagAwIBAgIBAjANBgkqhkiG9w0BAQQFADCBlzELMAkGA1UEBhMCSlAx..."
SSL_CLIENT_CERT_n	CA certificate starting from the CA that issues the SSL client certificate up to the root CA (n is a positive integer value displaying the number of chains) (DER-BASE64 format) The setting of the SSLExportCertChainDepth directive is required.	"MIIDrTCCAxagAwIBAgIBAjANBgkqhkiG9w0BAQQFADCBlzELMAkGA1UEBhMCSlAx..."
SSL_CLIENT_DN	Distinguish Name of the SSL client certificate subject	/C=JP/ST=Kanagawa/L=Yokohama/O=Hitachi/OU=soft/CN=c_name/EMAIL=c_name@soft.hitachi.co.jp
SSL_CLIENT_*Element*	Each element of the Distinguish Name of the SSL client certificate subject	Table B-6 shows an example of when SSL_CLIENT_DN is as shown above.
SSL_CLIENT_I_DN	Distinguish Name of the SSL client certificate issuer	/C=JP/ST=Kanagawa/L=Yokohama-shi/O=LOCAL-CA/OU=ca1/CN=ca1.hitachi.co.jp/EMAIL=ca-admin@ca1.hitachi.co.jp
SSL_CLIENT_I_*Element*	Each element of the Distinguish Name of the SSL client certificate issuer	Table B-7 shows an example of when SSL_CLIENT_I_DN is as shown above.

Table B-6: Example of SSL_CLIENT_ELEMENT

Environment variable name	Contents	Example
SSL_CLIENT_C	Country Name of the SSL client certificate subject	Jp
SSL_CLIENT_CN	Common Name of the SSL client certificate subject	c_name
SSL_CLIENT_EMAIL	E-Mail address of the SSL client certificate subject	c_name@soft.hitachi.co.jp
SSL_CLIENT_L	Locality Name of the of SSL client certificate subject	Yokohama
SSL_CLIENT_O	Organization Name of the SSL client certificate subject	Hitachi
SSL_CLIENT_OU	Organization Unit Name of the SSL client certificate subject	Soft
SSL_CLIENT_ST	State Name of the SSL client certificate subject	Kanagawa

Table B-7: Examples of SSL_CLIENT_I_ELEMENT

Environment variable name	Contents	Example
SSL_CLIENT_I_C	Country Name of the SSL client certificate issuer	JP
SSL_CLIENT_I_CN	Common Name of the SSL client certificate issuer	ca1.hitachi.co.jp
SSL_SERVER_I_EMAIL	E-Mail address of the SSL client certificate issuer	ca-admin@ca1.hitachi.co.jp
SSL_CLIENT_I_L	Locality Name of the SSL client certificate issuer	Yokohama-shi
SSL_CLIENT_I_O	Organization Name of the SSL client certificate issuer	LOCAL-CA
SSL_CLIENT_I_OU	Organization Unit Name of the SSL client certificate issuer	ca1
SSL_CLIENT_I_ST	State Name of the SSL client certificate issuer	Kanagawa

C. System monitoring with HA monitor - high availability system monitoring (operating a clustering system)

HA monitor, high availability system monitoring, is designed to switch systems including server programs (hereafter called "servers"), to improve system reliability and system utilization rates.

Cosminexus HTTP Server can be operated on a clustering system that uses HA monitor. For details on HA monitor, see the manual *High Availability System Monitoring Function HA Monitor*. For details on how to operate required software (such as the OS) and related programs (such as CGI programs) for Cosminexus HTTP Server in a clustering configuration environment, see the documentation for each program.

HA monitor allows you to operate Web server with minimum outage time of content delivery due to hardware failure and abnormal software stoppage. You can also administrate, maintain, and upgrade software without stopping services.

Main failures monitored

Monitored failures (failures detected by HA monitor) can be categorized into two types depending on where they occur: server failures and system failures . "System" here is a general term for the entire system, including hardware required for business processing as well as executed programs and communication devices. The main failures monitored by HA monitor are as follows:

Type of failure	Failure description
Server failure	• Logical error on the server
System failure	• Resource failure (for example, disk device failure) • Hardware failure in the system, or power down • Kernel failure • HA monitor failure • Monitored path failure • System slowdown

C.1 Example of hardware configuration and overview of HA monitor behavior

When a failure occurs in the system being monitored (hereafter called the *active system*), HA monitor switches to the secondary system (hereafter called the *standby system*) to continue processing. This functionality is called *system switching*.

The following section shows examples of hardware configuration and gives an outline of the operation.

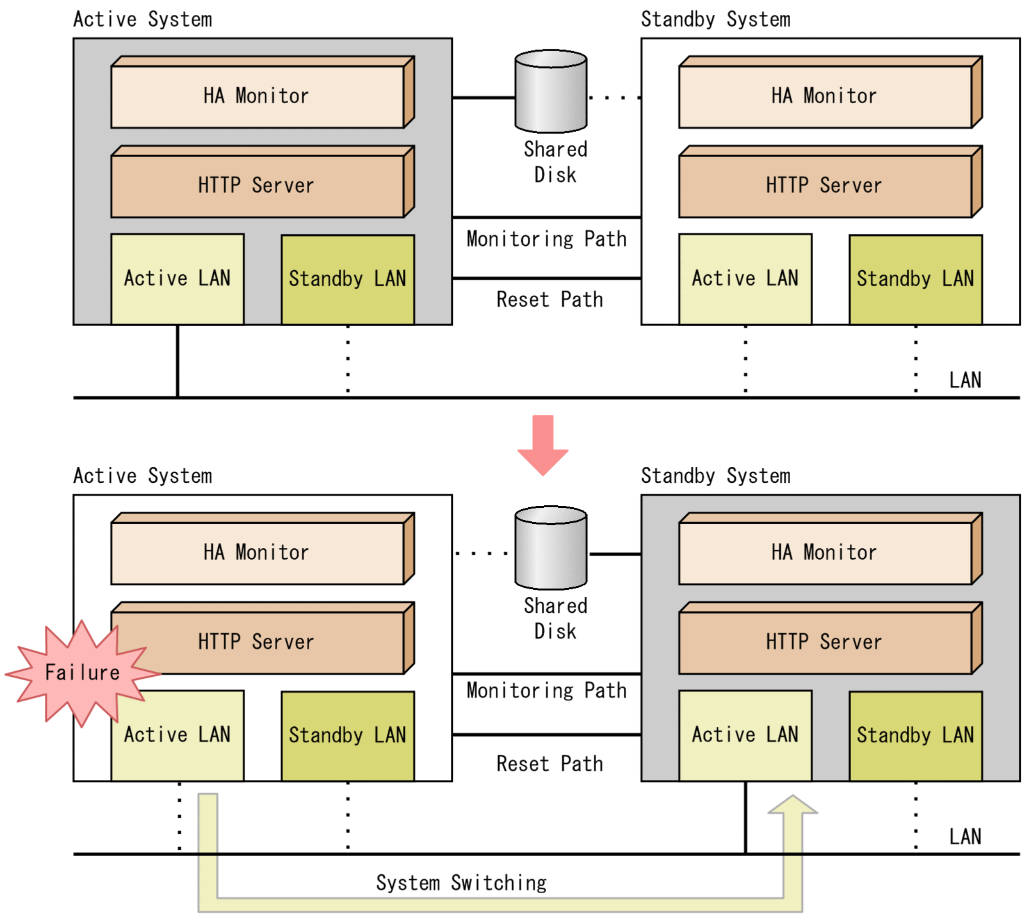
(1) Example of a one-to-one system switching configuration

The following example shows a one-to-one system configuration consisting of an active system and a standby system.

The configuration consists of active and standby systems connected by a LAN, in which servers provide services, monitoring paths for the systems to monitor each other, and a reset path for indicating a reset when a failure occurs in the active system. A disk storage device is shared between nodes.

When a failure occurs in the active system, HA monitor stops the active system and switches systems. The shared disk will be mounted onto the standby system. The following figure shows an example of the one-to-one system switching configuration.

Figure C-1: Example of the one-to-one system switching configuration

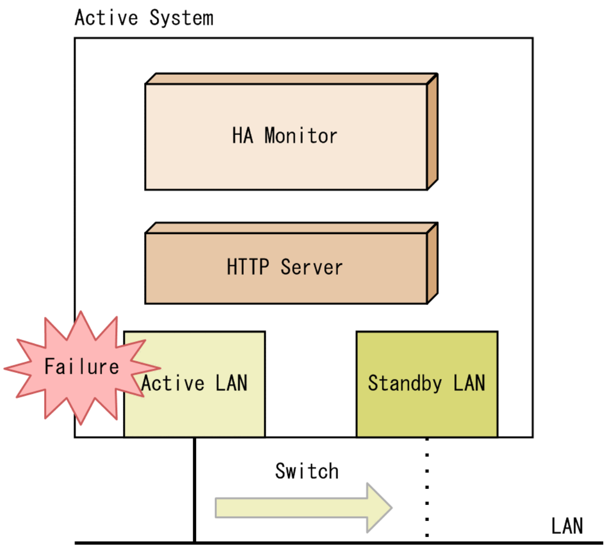


(2) Example of a double LAN adapter configuration

HP-UX HA monitor can control a double LAN adapter configuration in the system. You can define the LAN adapter used by the servers as a redundant model consisting of an active system and standby system. With this configuration, HA monitor can check the state of the LAN at intervals and if the active LAN adapter fails, HA monitor automatically switches to the standby LAN adapter.

The following figure shows an example of the double LAN adapter configuration

Figure C-2: Example of the double LAN adapter configuration



C.2 Cosminexus HTTP Server settings

To specify Cosminexus HTTP Server settings for HA monitor, follow the steps below:

1. Install Cosminexus HTTP Server onto the local disks of each system.
2. Create a configuration file for Cosminexus HTTP Server and distribute it onto each system.

Note the following points when setting up the server.

(1) For a virtual host

As a result of system switching, the server name returned to the client might change. Therefore, be sure to set the `ServerName` directive on virtual hosts.

(2) Specifying IP addresses

Instead of a physical IP address, use a logical IP address (alias IP address) for the directives that need an IP address to be specified (`<VirtualHost>`, `BindAddress`, `Listen`, and `NameVirtualHost`.)

(3) Checking the configuration file syntax

Before starting the HA monitor, run `/opt/hitachi/httpsd/sbin/httpsdctl configtest` to make sure that the server settings are correct.

(4) Editing the configuration file

By running `httpsdctl restart` or `httpsdctl graceful` directly from the command line, you can change the settings of Cosminexus HTTP Server while HA monitor is in use. Your changes need to be applied to the other system as well.

(5) Operation by using CRL

When operating by using CRL, the same type of CRL as the one being used for the active system needs to be set for the standby system.

C.3 Creating monitoring commands

In HA monitor, a program needs to be registered that notifies HA monitor of failures of a server that does not have an interface with HA monitor. Therefore, you need to create commands to monitor Cosminexus HTTP Server activities when you want Cosminexus HTTP Server to be monitored even though it does not have interface with HA monitor. You do not need to create such commands if the server is not to be monitored.

In Cosminexus HTTP Server, the execution commands differ from the processes that actually provides services. For a server to be monitored by HA monitor, define commands that can actually monitor the processes.

The following example shows how to write a shell script for monitoring Cosminexus HTTP Server behavior. When a failure occurs in Cosminexus HTTP Server, the shell script stops the process, and terminates its execution at the same time.

(Example)

The shell script `httpsd_monitor` monitors the process IDs stored in the file specified with the `PidFile` directive, and every five seconds checks whether the processes are running.

```
#!/bin/sh
###############################################################
#
### ALL RIGHTS RESERVED. COPYRIGHT (C) 2000, 2002, HITACHI,LTD.
###############################################################
#
HWSIDFILE=/opt/hitachi/httpsd/logs/httpd.pid
HWSITIME=5

if [ ! -e $HWSIDFILE ]
```

```
then
    exit 1
fi

HWSID=`cat $HWSIDFILE`
if [ x$HWSID = "x" ]
then
    exit 1
fi

while true
do
   STATUS=`ps -p $HWSID | grep $HWSID | awk '{print $1}' `
   if [ x$STATUS = "x" ]
   then
     break
   fi
   sleep $HWSITIME
done

exit 0
```

(1) Notes

Cosminexus HTTP Server provides a process that controls a group of processes that handle requests (see *4.1 Relationship between processes and directives of Cosminexus HTTP Server*.) In the example above, the script `httpsd_monitor` monitors whether the control process is running. The script does not monitor the behavior of the processes that handle requests.

C.4 HA monitor settings

This section explains how to specify HA monitor settings for Cosminexus HTTP Server and related programs if needed. For more information and points not covered here, see the manual *HA Monitor*.

To specify HA monitor settings for Cosminexus HTTP Server.

1. Configure the HA monitor environment.
2. Create a script that monitors Cosminexus HTTP Server, as well as start and stop scripts, and distribute them to the systems as necessary.
3. Configure the environment for the HA monitor that is associated with Cosminexus HTTP Server.
4. Start HA monitor. Then, use HA monitor commands to start Cosminexus HTTP Server.

(1) Creating start and stop scripts

You need to create and store start and stop scripts for HA monitor to be able to start and stop Cosminexus HTTP Server.

(a) Example of a start script

```
#!/bin/sh
/opt/hitachi/httpsd/sbin/httpsdctl start
```

(b) Example of a stop script

```
#!/bin/sh
/opt/hitachi/httpsd/sbin/httpsdctl stop
```

(2) Configuring the environment for HA monitor that supports Cosminexus HTTP Server

In the file `servers` under the directory for the HA monitor environment configuration, set up the server configuration. In this file, you can set up a start script, stop script, and monitoring commands.

(a) Example of an environment configuration

An example of an environment configuration for servers is given below. For parameters and details on the settings, see the manual *HA Monitor*.

```
/* example of environment configuration for servers (example for active system) */
       server  name                  /home/work/hws-start.sh, /* start script */
               alias                 HWS,
               acttype               monitor,
               termcommand           /home/work/hws-stop.sh,  /* stop script */
               switchtype            switch,
               initial               online,                  /* configuration for
active system */
             patrolcommand        /home/work/httpsd_monitor, /* monitoring command */
               servexec_retry        2,
               waitserv_exec         yes;
```

(3) Notes

When an active system is switched to another system, normal HTTP connections and connections using SSL are all disconnected, and these are not taken over by the standby system. In this case, the clients need to be reconnected.

D. System monitoring with MC/ServiceGuard (operating a cluster)

MC/ServiceGuard is a Hewlett-Packard software product for building clustering systems. Cosminexus HTTP Server can be operated on a clustering system using MC/ServiceGuard. For details on MC/ServiceGuard, see the MC/ServiceGuard documentation. Also, for details on how to operate required software (such as the OS) and related programs (such as CGI programs) for Cosminexus HTTP Server in a clustering configuration environment, see the documentation for each program.

MC/ServiceGuard allows you to operate a Web server with minimum content-delivery outage times caused by hardware failures or abnormal software stoppages. You can also administer, maintain, and upgrade software without stopping services.

Main failures monitored

The main failures monitored by MC/ServiceGuard are as follows:

- LAN failures
- Resource failures (system processor, disk, interface)
- Abnormal software stops

D.1 Example of hardware configuration and overview of MC/ServiceGuard behavior

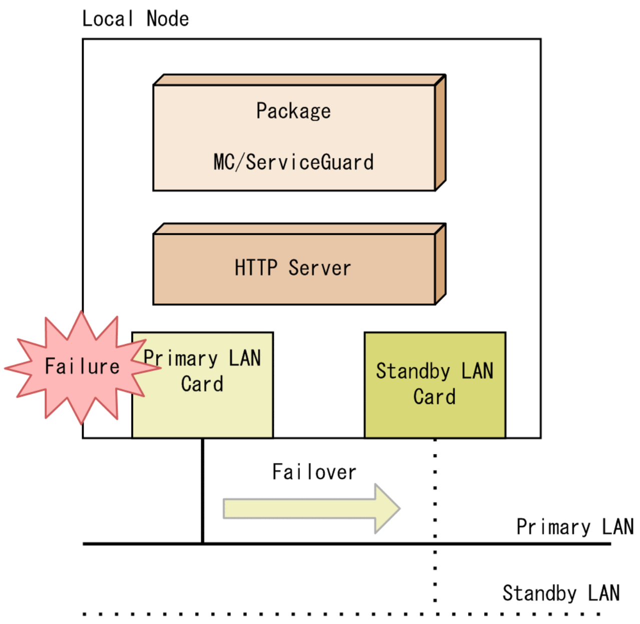
When MC/ServiceGuard detects a failure in a monitored system (hereafter called the *primary system*), it switches processes to the secondary system (hereafter called the *standby system*) to continue services. This function is called *failover*. Behavior of a failover differs depending on whether the mode is in local node operation or multiple node operation. Examples of hardware configuration and an overview of failover for the respective operations are as follows.

(1) Example of local node operation

The following example shows a double LAN configuration, one LAN is the primary LAN and the other is the standby LAN. Supported LAN cards are connected to each LAN.

In this configuration, when a failure occurs in the primary LAN card, the connection will be switched to the standby LAN card that resides on the same node. The following figure shows an example of local node operation.

Figure D-1: Example of local node operation



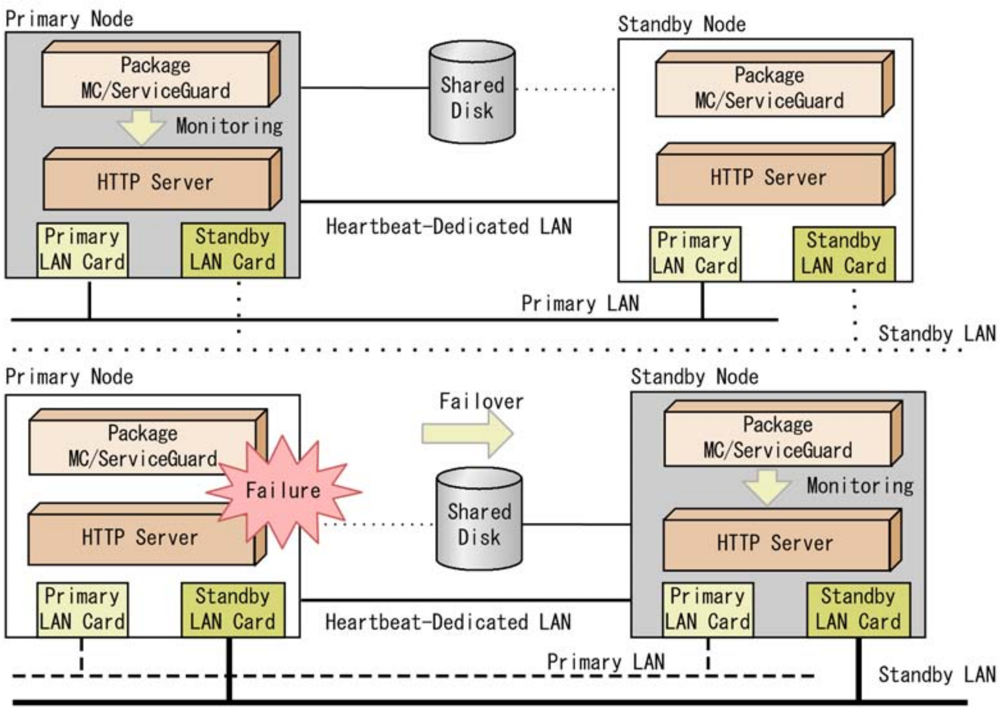
(2) Example of multiple node operation

The example below shows a double LAN configuration.

One LAN is the primary LAN and the other is the standby LAN, and supported LAN cards are connected to each node. This is a double LAN communication line configuration consisting of either a heartbeat-dedicated LAN or a heartbeat line using RS232 signals, in addition to the primary LAN. The disk storage device is shared between nodes.

If a failure occurs in the primary node and MC/ServiceGuard determines that a failover to another node is needed, MC/ServiceGuard stops the primary system and starts the package on the standby system to continue services. The shared disk is mounted on the standby system. The following figure shows an example of multiple node operation.

Figure D-2: Example of multiple node operation (MC/ServiceGuard)



D.2 Cosminexus HTTP Server settings

To specify Cosminexus HTTP Server settings for MC/ServiceGuard, follow the steps below.Install Cosminexus HTTP Server onto a local disk at each node.

1. Create a script for monitoring Cosminexus HTTP Server, as needed.

   For details on creating scripts, see *Appendix D.3 Creating a monitoring script*.

2. Set up MC/ServiceGuard.

   For details on setting up MC/ServiceGuard, see *Appendix D.4 MC/ServiceGuard settings*.

3. To each node, distribute the Cosminexus HTTP Server configuration file, the monitoring script that you created in the previous step, and the script for controlling MC/ServiceGuard packages.

4. Start MC/ServiceGuard.

Note the following points when configuring environments.

(1) For a virtual host

As a result of a failover, the server name returned to the client might change. Therefore, be sure to set the `ServerName` directive on virtual hosts.

(2) Specifying IP addresses

Instead of a stationary IP address (which cannot be moved to another node), use a relocatable IP address (which is assigned to a package and can be moved to another node) for the directives that need an IP address to be specified (`<VirtualHost>`, `BindAddress`, `Listen`, and `NameVirtualHost`.)

(3) Checking the configuration file syntax

Before starting MC/ServiceGuard, run `/opt/hitachi/httpsd/sbin/httpsdctl configtest` to make sure that the server settings are correct.

(4) Editing the configuration file

By running `httpsdctl restart` or `httpsdctl graceful` directly from the command line, you can change the settings of Cosminexus HTTP Server while MC/ServiceGuard is in use. Your changes need to be applied to the other nodes as well.

(5) Operation by using CRL

In the standby node, you need to set the same type of CRL as the one being used for the primary node.

D.3 Creating a monitoring script

In MC/ServiceGuard, to enable software to be monitored, the execution command must be the same as the actual service name, and that process must be running until the service ends.

In Cosminexus HTTP Server, the execution commands differ from the processes that actually provides services. For a server to be monitored by MC/ServiceGuard, create a script that can actually monitor the processes.

However, if a server is not monitored, or operation is only within the local node where no process will fail over to another node, you do not need to create scripts.

The following example shows how to write a shell script for monitoring Cosminexus HTTP Server behavior. When a failure occurs in Cosminexus HTTP Server, the shell script stops the process, and terminates its execution at the same time.

(Example)

The shell script `httpsd_monitor` monitors the process IDs stored in the file specified with the `PidFile` directive, and every five seconds checks whether the process is running. Specify the `PidFile` directive value in the form of an absolute path as an argument.

```
#!/bin/sh
```

```
HWSITIME=5
if [ $# -ne 1 ]
then
    exit 1
fi

HWSIDFILE=$1

if [ ! -e $HWSIDFILE ]
then
    exit 1
fi

HWSID=`cat $HWSIDFILE`
if [ x$HWSID = "x" ]
then
    exit 1
fi

while true
do
   STATUS=`ps -p $HWSID | grep $HWSID | awk '{print $1}' `
   if [ x$STATUS = "x" ]
   then
     break
   fi
   sleep $HWSITIME
done

exit 0
```

(1) Notes

Cosminexus HTTP Server provides a process that controls a group of processes that handle requests. (see *4.1 Relationship between processes and directives of Cosminexus HTTP Server*.) In the example above, the script `httpsd_monitor` monitors whether the control process is running. The script does not monitor the behavior of the processes that handle requests.

D.4 MC/ServiceGuard settings

Define Cosminexus HTTP Server, and related programs as necessary, for the packages. For more information and points not covered here, see the MC/ServiceGuard documentation.

(1) Cluster configuration and package configuration

Examples of a cluster configuration and package configuration are as follows.

(a) Example of a cluster configuration

```
CLUSTER_NAME cluster1
FIRST_CLUSTER_LOCK_VG /dev/vg01
NODE_NAME original_node
NETWORK_INTERFACE lan0
HEARTBEAT_IP 172.16.1.1
FIRST_CLUSTER_LOCK_PV /dev/dsk/c1t2d0
NODE_NAME adoptive_node
NETWORK_INTERFACE lan0
HEARTBEAT_IP 172.16.1.2
FIRST_CLUSTER_LOCK_PV /dev/dsk/c1t2d0
HEARTBEAT_INTERVAL 1000000
NODE_TIMEOUT 2000000
AUTO_START_TIMEOUT 600000000
NETWORK_POLLING_INTERVAL 2000000
MAX_CONFIGURED_PACKAGES 10
VOLUME_GROUP /dev/vg01
```

(b) Example of a package configuration

```
PACKAGE_NAME CosminexusHTTPServer
FAILOVER_POLICY CONFIGURED_NODE
FAILBACK_POLICY MANUAL
NODE_NAME original_node
NODE_NAME adoptive_node
RUN_SCRIPT /etc/cmcluster/CosminexusHTTPServer/control.sh
RUN_SCRIPT_TIMEOUT NO_TIMEOUT
HALT_SCRIPT /etc/cmcluster/CosminexusHTTPServer/control.sh
HALT_SCRIPT_TIMEOUT NO_TIMEOUT
SERVICE_NAME httpsd_check
SERVICE_FAIL_FAST_ENABLED NO
SERVICE_HALT_TIMEOUT 300
SUBNET 172.16.1.0
PKG_SWITCHING_ENABLED YES
NET_SWITCHING_ENABLED YES
NODE_FAIL_FAST_ENABLED NO
```

(2) Writing a package control script

This section explains how to write a package control script when you want Cosminexus HTTP Server to be monitored. For cases other than those described below, the settings differ based on the system.

(a) Storing a script

Store the shell script `httpsd_monitor` you have created as a service to be monitored by MC/ServiceGuard. The following example assumes that `httpsd_monitor` is stored in `/opt/hitachi/httpsd/bin`.

Note that the same value for the `PidFile` directive specified in Cosminexus HTTP

Server configuration file must be specified in the `httpsd_monitor` argument. You do not need to store `SERVER_NAME` and `SERVER_CMD` if the Cosminexus HTTP Server is not to be monitored.

```
PATH=/sbin:/usr/bin:/usr/sbin:/etc:/bin
VGCHANGE="vgchange -a e"
VG[0]=/dev/vg01
LV[0]=/dev/vg01/lvol1
FS[0]=/MCSG
FS_MOUNT_OPT[0]="-o rw"
IP[0]=172.16.1.3
SUBNET[0]=172.16.1.0
SERVICE_NAME[0]="httpsd_check"
SERVICE_CMD[0]="/opt/hitachi/httpsd/bin/httpsd_monitor
                /opt/hitachi/httpsd/logs/httpd.pid"
SERVICE_RESTART[0]="-r 0"
```

(b) Defining a function

Define processes to start or stop Cosminexus HTTP Server in the function `customer_defined_run_cmds` in the package control script (to start the package) or in the function `customer_defined_halt_cmds` (to stop the package).

To start:

```
function customer_defined_run_cmds
{
# ADD customer defined run commands.
: # do nothing instruction, because a function must contain some
command.
  /opt/hitachi/httpsd/sbin/httpsdctl start
  test_return 51
}
```

To stop:

```
function customer_defined_halt_cmds
{
# ADD customer defined halt commands.
: # do nothing instruction, because a function must contain some
command.
  /opt/hitachi/httpsd/sbin/httpsdctl stop
test_return 52
}
```

(3) Notes

When a failover to another node occurs, normal HTTP connections and connections using SSL are all disconnected, and these are not taken over by the standby node. In this case, the clients need to be reconnected.

E. System monitoring with HACMP for AIX (operating Cluster Multi-Processing)

HACMP for AIX is a software product developed by IBM, for building a mission-critical computing platform. HACMP for AIX has two main components, High Availability (HA) and Cluster Multi-Processing (CMP). Cosminexus HTTP Server can be operated through Cluster Multi-Processing by using HACMP for AIX. For details on HACMP for AIX, see the HACMP for AIX documentation. Also, for details on how to operate required software (such as the OS) and related programs (such as CGI programs) for Cosminexus HTTP Server in a Cluster Multi-Processing (CMP) configuration, see the manuals for each program.

HACMP for AIX allows you to operate a Web server with minimum content-delivery outage times caused by hardware failures or abnormal software stoppages. You can also administer, maintain, and upgrade software without stopping services.

Main failures monitored

The main failures monitored by HACMP for AIX are as follows:

- LAN failures
- Resource failures (system processor, disk, interface)
- Abnormal software stops

E.1 Example of hardware configuration and overview of HACMP for AIX behavior

When HACMP for AIX detects a failure in a monitored node, it switches processes to the standby node to continue services. This behavior is called a *takeover*. A takeover has the following switching functionalities:

- Node
- Application
- Network and network adapter
- Disk and disk adapter

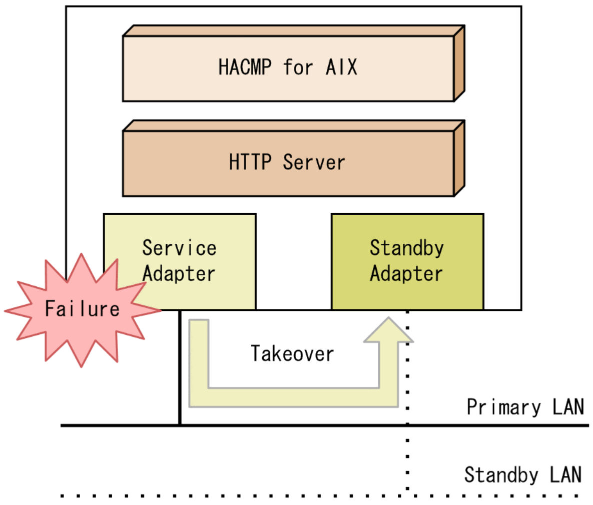
Examples of a network failure and application failure are as follows.

(1) Example of a network failure

In a double LAN adapter configuration, define one adapter to run for application services (as the *service adapter*), and the other as a backup service adapter (as the *standby adapter*).

In this configuration, when a failure occurs in the service adapter, the connection will be switched to the standby adapter that resides on the same node. The following figure shows an example of double adapter configuration.

Figure E-1: Double adapter configuration



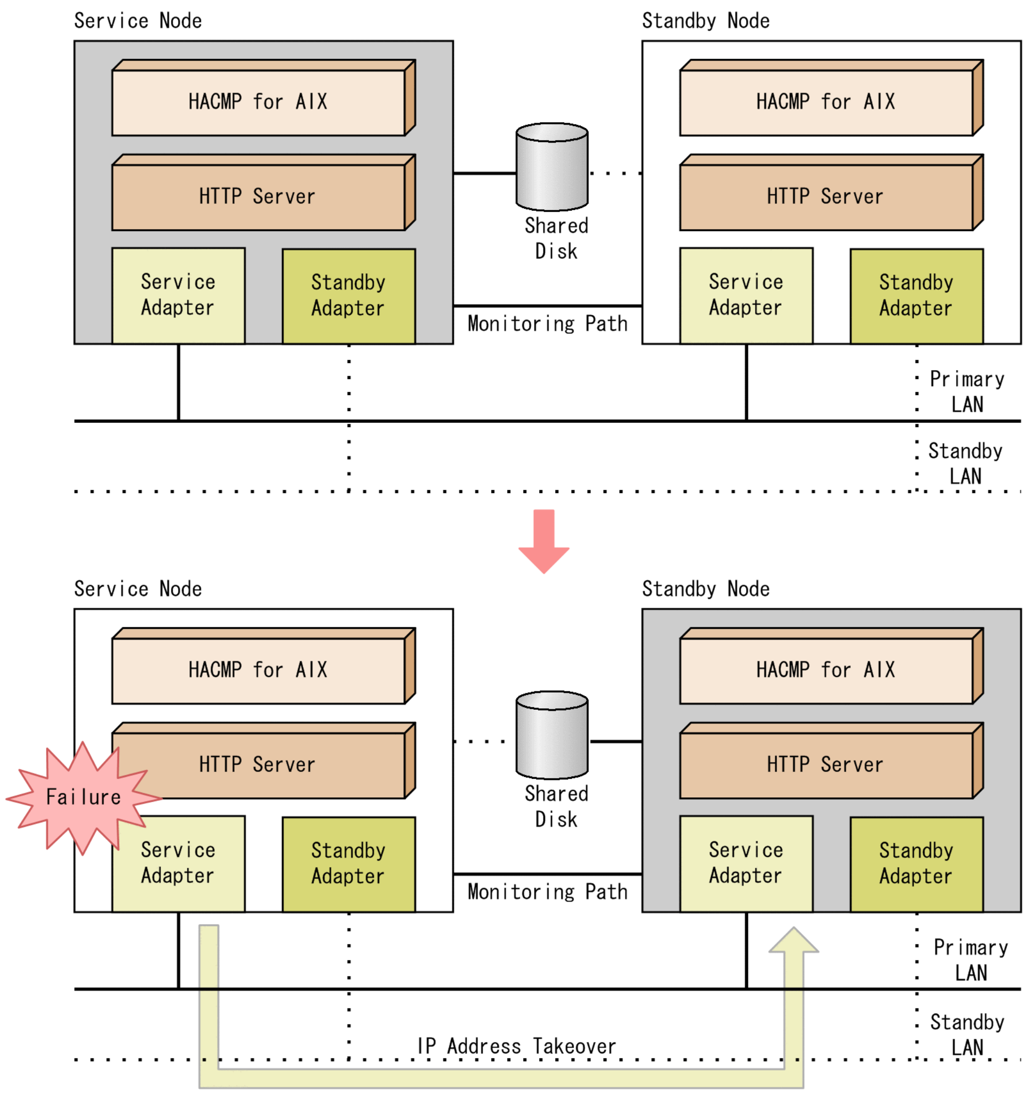
(2) Example of an application failure

The following example shows a double node configuration.

The example is a double LAN configuration consisting of one active node (service node) and one backup node (standby node), where the two nodes are connected using RS232 signals as a monitoring path. The disk storage device is shared between nodes.

If a failure occurs in the service node and HACMP for AIX determines that a failover to another node is needed, HACMP for AIX removes the service node and continues the services on the standby node. The shared disk can continue to be used on the standby node. The following figure shows an example of multiple node operation.

Figure E-2: Example of multiple node operation (HACMP for AIX)



E.2 Cosminexus HTTP Server settings

To specify Cosminexus HTTP Server settings for HACMP for AIX, follow the steps below.

1. Install Cosminexus HTTP Server onto the local disks of each node.
2. On the service node, create a Cosminexus HTTP Server configuration file, start script, stop script, and monitor method.

3. Distribute the configuration file, start script, stop script, and monitor method to the standby node, as needed.
4. In HACMP for AIX , define the application server used for Cosminexus HTTP Server.
5. In HACMP for AIX, define the application monitor used for Cosminexus HTTP Server.
6. Complete the definitions for HACMP for AIX, and sync the cluster definition in all nodes.
7. Start the cluster service.

Note the following points when configuring environments.

(1) Checking the configuration file syntax

Before starting cluster service, run `/opt/hitachi/httpsd/sbin/httpsdctl configtest` to make sure that the server settings are correct.

(2) Editing the configuration file

By running `httpsdctl restart` or `httpsdctl graceful` directly from the command line, you can change the settings of Cosminexus HTTP Server while it is in use on HACMP for AIX. Your changes need to be applied to the other nodes as well.

(3) Operation with CRL

In the standby node, you need to set the same type of CRL as the one being used for the service node.

E.3 Creating a monitoring script

When you want Cosminexus HTTP Server to be monitored by HACMP for AIX, you need to create a script that monitors Cosminexus HTTP Server and store it in the monitor method. This script must return 0 when Cosminexus HTTP Server is working normally and return a value other than zero when a problem is detected.

In Cosminexus HTTP Server, the execution command differs from the process that actually provides services. For the server to be monitored by HACMP for AIX, create a script that can actually monitor the process.

However, if the server is not monitored, or operation is only within the local node, you do not need to create scripts.

The following example shows how to write a shell script for monitoring Cosminexus HTTP Server behavior. The sample script returns 0 when Cosminexus HTTP Server is working normally, and returns a value other than zero when a problem is detected.

(Example)

The script returns 0 if a process ID stored in the file specified with the `PidFile` directive is running, and returns 1 if it is not running.

```
#!/bin/sh


HWSIDFILE=/opt/hitachi/httpsd/logs/httpd.pid
if [ ! -e $HWSIDFILE ]
then
        exit 1
fi

HWSID=`cat $HWSIDFILE`
if [ x$HWSID = "x" ]
then
        exit 1
fi

STATUS=`ps -p $HWSID | grep $HWSID | awk '{print $1}'`

if [ x$STATUS = "x" ]
then
        exit 1
else
        exit 0
fi
```

(1) Notes

Cosminexus HTTP Server provides a process that controls a group of processes that handle requests (see *4.1 Relationship between processes and directives of Cosminexus HTTP Server*). In the example above, the script monitors whether the control process is running. The script does not monitor the behavior of the processes that handle requests.

E.4 HACMP for AIX settings

Define Cosminexus HTTP Server, and related programs as necessary, for the packages. For more information and points not covered here, see the HACMP for AIX manuals.

(1) How to register Cosminexus HTTP Server as an application server

To manage Cosminexus HTTP Server in HACMP for AIX, you need to register Cosminexus HTTP Server as an application server.

Open the SMIT utility and select the **Add and Application Server** window to register the server name for Cosminexus HTTP Server, start script, and stop script.

(a) Example of a start script

```
#!/bin/sh
/opt/hitachi/httpsd/sbin/httpsdctl start
```

(b) Example of a stop script

```
#!/bin/sh
/opt/hitachi/httpsd/sbin/httpsdctl stop
```

(2) How to monitor Cosminexus HTTP Server

If you want Cosminexus HTTP Server to be monitored by HACMP for AIX, you need to store a script. Create a script that best suits your operating environment.

If you are using HACMP/ES (Enhanced Scalability feature), open the SMIT utility and select the **Add Custom Application Monitor** window to store the monitoring script for Cosminexus HTTP Server.

(3) Notes

When takeover to another node occurs, normal HTTP connections and connections using SSL are all disconnected, and these are not taken over by the standby node. In this case, the clients must be reconnected.

F. System monitoring with Windows Server Failover Cluster

Windows Server Failover Cluster is a software product of Microsoft. Cosminexus HTTP Server can use Windows Server Failover Cluster to run and operate cluster services. For details on Windows Server Failover Cluster, see the Windows Server Failover Cluster documentation.

The main failures monitored by server clusters are as follows:

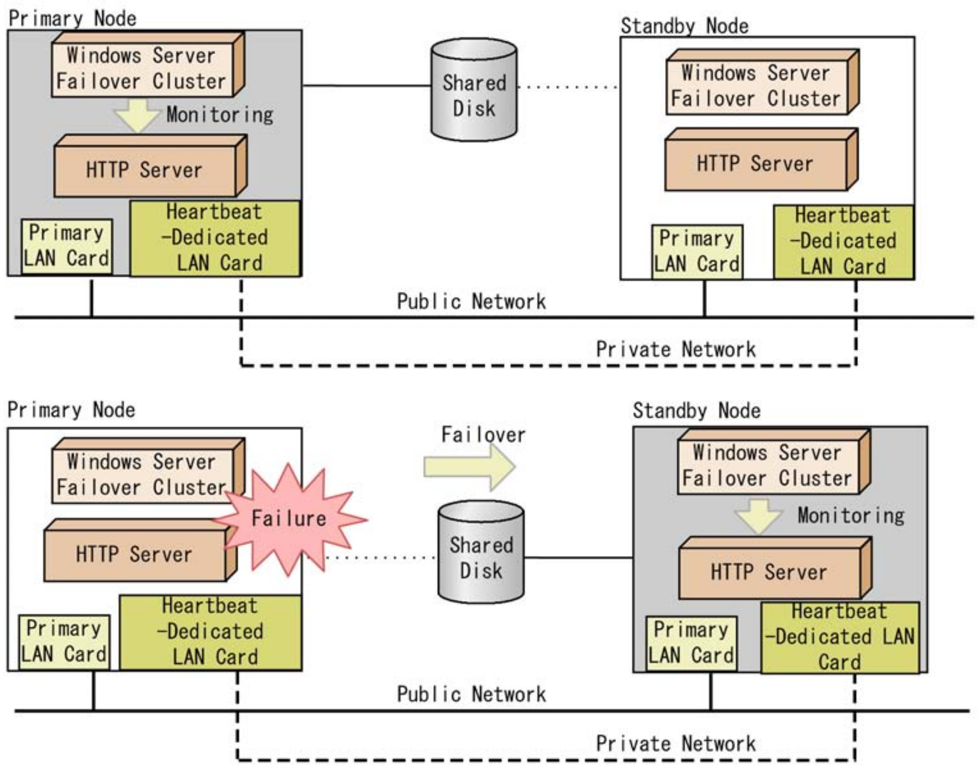
- LAN failures
- Resource failures (system processor, disk, interface)
- Abnormal software stops

When a server cluster detects a failure in the monitored system (hereafter called the *primary system*), it switches processes to the secondary system (hereafter called the *standby system*) to continue services. This function is called *failover*.

F.1 Example of operation

If a failure occurs in the primary node and HACMP for AIX determines that a failover to another node is needed, HACMP for AIX stops the primary system and starts services on the standby system to continue. The shared disk will be mounted onto the standby system. The following figure shows an example of multiple node operation.

Figure F-1: Example of multiple node operation (Windows Server Failover Cluster)



F.2 Cosminexus HTTP Server settings

To set up Cosminexus HTTP Server for server cluster, follow the steps below:

1. Install Cosminexus HTTP Server onto the local disk for each node.
2. On each node, register Cosminexus HTTP Server as a Windows service. The service name to register must be the same across nodes. You can skip this step if you are clustering services named *Cosminexus HTTP Server* which are stored when Cosminexus HTTP Server is installed.
3. Set up the server cluster.

   See *Appendix F.3 Server cluster settings*.
4. Distribute the Cosminexus HTTP Server configuration file to each node.
5. Place the clustering services and applications in the server cluster online.

Note the following points when configuring environments.

(1) For a virtual host

As a result of a failover, the server name returned to the client might change. Therefore, be sure to set the `ServerName` directive on virtual hosts.

(2) Specifying IP addresses

Instead of an IP address specified in the LAN card (which is not portable to another node), use a relocatable IP address (which is assigned to a resource module, and is portable to another node) for the directives that need an IP address to be specified (`<VirtualHost>`, `BindAddress`, `Listen`, and `NameVirtualHost`.)

(3) Checking the configuration file syntax

Before starting the server cluster, run `httpsd -t` in the directory where it is installed to make sure that the server settings are correct. If resources belonging to the services or applications in the cluster (such as an IP address or memory space) are referenced in the Cosminexus HTTP Server configuration file, move the services and applications to the node you are checking, and then run `httpsd -t`.

(4) Editing the configuration file

You can change the settings of Cosminexus HTTP Server without placing offline the services being used by the server cluster, by restarting the Cosminexus HTTP Server services with the `httpsd` command or from the start menu, in the node on which the services are running online. Your changes to the configuration files need to be applied to the other nodes as well

(5) Operation by using CRL

In the standby node, you need to set the same type of CRL as the one being used for the primary node.

(6) Resource type

When you specify Cosminexus HTTP Server in the resources, specify it by universal services rather than by universal applications. Cosminexus HTTP Server does not work properly as a universal application.

F.3 Server cluster settings

You can specify the Windows Server Failover Cluster settings by using the cluster management software called Failover Cluster Manager and the `cluster` command. For details, see the Microsoft documentation.

To specify the settings, follow the steps below:

1. Use Failover Cluster Manager to create a clustering service for Cosminexus HTTP Server. Add resources that move between nodes at failover, such as general-purpose services belonging to clustering services (Cosminexus HTTP Server services), client access point (name and IP address), or storage area. Then

open properties for each item to change settings for resource dependencies and other details regarding clusters.

2. Run the command prompt as an administrator.
3. In the command prompt window, run the following command:

   `cluster res "`*resource name*`" /priv StartupParameters=""`For "*resource name*", specify the resource name of the Cosminexus HTTP Server general-purpose service. You can locate the resource name for the Cosminexus HTTP Server general-purpose services in **Failover Cluster Manager**.
4. From **Failover Cluster Manager**, open the properties of the Cosminexus HTTP Server general-purpose services and make sure that the value for **Setup Parameter** is blank.

G. Notes on migration from earlier versions

Note:

If Hitachi Web Server is installed, first uninstall it and then install Cosminexus HTTP Server. Do not perform an overwrite installation over an older version of Hitachi Web Server.

The following table lists points to note when migrating from an older version, and indicates whether settings need to be changed.

Table G-1: Points to note when migrating from an older version

No.	Item	Cosminexus Application Server version before the migration		
		V6	V7	V8
1	Specification of the `ErrorDocument` directive string	Y	-	-
2	Specification of a reverse proxy	Y	-	-
3	Specification of the `ShmemUIDisUser` directive	Y	-	-
4	If programs other than `rotatelogs` or `rotatelogs2` are specified in the pipe format for the `TransferLog`, `CustomLog`, and `ErrorLog` directives (Windows version only)	Y	-	-
5	Change in behavior when a SIGXFSZ or SIGXCPU signal is received (UNIX version only)	Y	Y	-
6	Change in the update interval when the activity status is displayed (status information display)	Y	Y	-
7	GUI server manager functionality is no longer provided	Y	Y	-
8	The HTML manual is no longer provided	Y	Y	-
9	Change of program product name	Y	Y	Y
10	The program menu is no longer provided (Windows version only)	Y	Y	Y
11	Change of SSL related commands	Y	Y	Y
12	The SSLv2 protocol is not supported	Y	Y	Y
13	Change in supported encryption types	Y	Y	Y
14	Specification at SSL client authentication	Y	Y	Y

(Legend)

- V6, V7, and V8 stand for Cosminexus Application Server Version 6, Version 7, and Version 8, respectively
- Items for which settings need to be changed are as follows:

  Y: Settings need to be changed at migration

  -: No settings need to be changed

The following are points to note for each item. If settings need to be changed, perform the tasks from installation to startup.

1. Specification of the `ErrorDocument` directive textFor Cosminexus Application Server Version 6, the specification added a double quotation mark (") at the beginning of the specification string. In this version, however, enclose the string in double quotation marks ("...")

   (Example)

   In Cosminexus Application Server Version 6:

```
ErrorDocument 500 "Server Error.
```

   In this version:

```
ErrorDocument 500 "Server Error."
```

2. Specification of a reverse proxy:For Cosminexus Application Server Version 6, only `mod_proxy.so` (for the Windows version) or `libproxy.so` (for the UNXI version) was loaded. In this version, however, load both `mod_proxy.so` and `mod_proxy_http.so`.

   For the UNIX version, you must always load them in the order shown in the following example.

   (Example)

   For the Windows version:

   - In Cosminexus Application Server Version 6

```
LoadModule proxy_module modules/mod_proxy.so
```

   - In this version

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so
```

   For the UNIX version:

- In Cosminexus Application Server Version 6

```
LoadModule proxy_module libexec/libproxy.so
```

- In this version

```
LoadModule proxy_module libexec/mod_proxy.so
LoadModule proxy_http_module libexec/mod_proxy_http.so
```

3. Specification of the `ShmemUIDisUser` directive The `ShmemUIDisUser` directive is not needed for this version. Remove this directive specification.
4. If programs other than `rotatelogs` or `rotatelogs2` are specified in the pipe format for the `TransferLog`, `CustomLog`, and `ErrorLog` directives (Windows version only)

   The code for linefeeds in the log information has been changed from LF to CRLF. So some programs might need to be revised accordingly.

   When you specify a program for a directive, you need to add the program extension (for example, `.exe`).

   (Example)

   To specify a user-created program `writelogs.exe` to output log information:

   - In Cosminexus Application Server Version 6

```
CustomLog "|\"\"C:/proprietary/writelogs\" program argument\""
```

   - In this version

```
CustomLog "|\"\"C:/proprietary/writelogs.exe\" program argument \""
```

5. Change in behavior when a SIGXFSZ or SIGXCPU signal is received (UNIX version only)

   At the reception of the SIGXFSZ signal or SIGXCPU signal, the control process now stops Web server.
6. Change in the update interval when the activity status is displayed (status information display)

   For the interval for updating the status information (specified in `refresh=`*update-interval*), the specifiable interval range was changed from 1 to 3,600 seconds. Also the default value was changed to 60 seconds.
7. The GUI server manager functionality is no longer provided.

   The GUI server manager functionality is no longer supported.

8. The HTML manual is no longer provided

   The shipment does not include the HTML manual.

9. Change of program product name

   The name of the program product has been changed from *Hitachi Web Server* to *Cosminexus HTTP Server*. Accordingly, the names used for logs or HTTP communications have changed from *Hitachi Web Server* to *Cosminexus HTTP Server*. In the Windows version, the service name created by default has also been changed, except when the service *Hitachi Web Server* already exists in your environment from an installation of an earlier version.

10. The program menu is no longer provided (Windows version only)

    The program menu of this product is no longer created in the Windows **Start** menu.

11. Change of SSL related commands

    The `sslc` (or `sslckey`) and `sslccert` commands have been changed to the `keygen` and `certutil` commands. Note that the following functions run by the `sslc` command are not included in this change.

    - Creating private keys and certificates for a test CA (Certification Authority), and signing at a test CA
    - Viewing the content of CRL and changing its format

    For details on the `keygen` and `certutil` commands, see *5.2 Acquiring a certificate*.

    You can continue to use key and certificate files you created with the old commands. However, you cannot use new commands together with an old command (for example, the `sslckey` or `certutil` command) to create key or certificate files.

12. The SSLv2 protocol is not supported

    SSLv2 can no longer be specified for the `SSLProtocol` directive. If you send a request from a client that supports SSLv2 only, the SSL handshake will result in an error and the connection will fail.

13. Change in supported encryption types

    Supported encryption types have been changed. For details, see *6.2.7(20) SSLBanCipher encryption-type [encryption-type ...]*.

14. Specification at SSL client authentication

    You can no longer specify 3 for the `SSLVerifyCLient` directive.

    When implementing SSL client authentication, in the `SSLCACertificateFile` or `SSLCACertificatePath` directive, always specify a certificate from the CA who issued the client certificate (including the intermediate CA certificate and route CA certificate).

H. Glossary

Terminology used in this manual

For the terminology, see *Appendix M. Glossary* in the manual *uCosminexus Application Server Application Development Guide*.

Index