

# Tamino

## Communication with Tamino's X-Machine

Version 9.7

April 2015

This document applies to Tamino Version 9.7.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 1999-2015 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at <http://softwareag.com/licenses>.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at <http://softwareag.com/licenses/> and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at <http://softwareag.com/licenses> and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

**Document ID: INS-COMMUNICATIONS-97-20160318**

## Table of Contents

Communication with Tamino's X-Machine .....	v
1 Introduction .....	1
2 Communication Methods .....	3
Introduction .....	4
Communication via a Web Server .....	5
HTTP/HTTPS based communication .....	6
The "webserviceless" option of the Java or C API .....	8
3 Web Servers .....	9
4 Webserviceless APIs .....	11
5 Security .....	13
Introduction .....	14
Web Server Assignment .....	14
Encryption .....	14
Using SSL with Native HTTP access (HTTPS) .....	15
6 Request Handling in the Server .....	17
Index .....	19



---

# Communication with Tamino's X-Machine

---

This document describes the methods available to clients for communicating with Tamino's X-Machine.

This document is intended for use by application programmers who wish to develop client applications that communicate with the X-Machine.

This information is structured into the following sections:

*Introduction*

*Communication Methods*

*Web Servers*

*Webserverless APIs*

*Security*

*Request Handling in the Server*

---

# 1 Introduction

---

Clients that access the X-Machine can be tools (e.g. Tamino Interactive Interface or Tamino Data Loader), applications that use the Tamino APIs, or applications that use HTTP directly without a Tamino API.

Communication issues of other components (e.g. Tamino Manager) are not discussed here. Traditionally, the X-Machine is accessed via a web server; from Tamino 4.1, direct (*webserversless*) access is also possible. It makes no difference to the X-Machine whether requests are shipped with or without a web server.

The user's session state in the database is not related to any communication machinery. Requests relating to a single database session can be issued from various machines via various transport layers, as long as they contain the appropriate session ID and session key.



# 2 Communication Methods

---

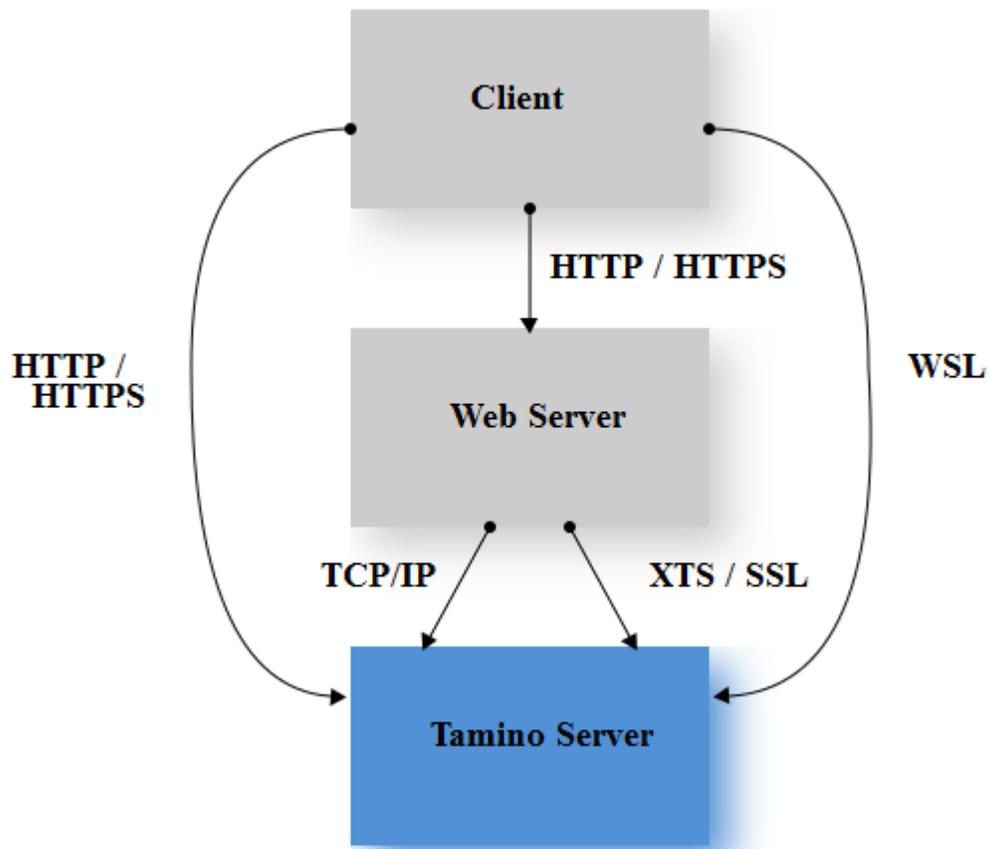
- Introduction ..... 4
- Communication via a Web Server ..... 5
- HTTP/HTTPS based communication ..... 6
- The "webserviceless" option of the Java or C API ..... 8

## Introduction

---

Tamino offers three communication methods:

- via a web server,
- via direct HTTP resp. HTTPS, or
- WSL (*webserversless*)

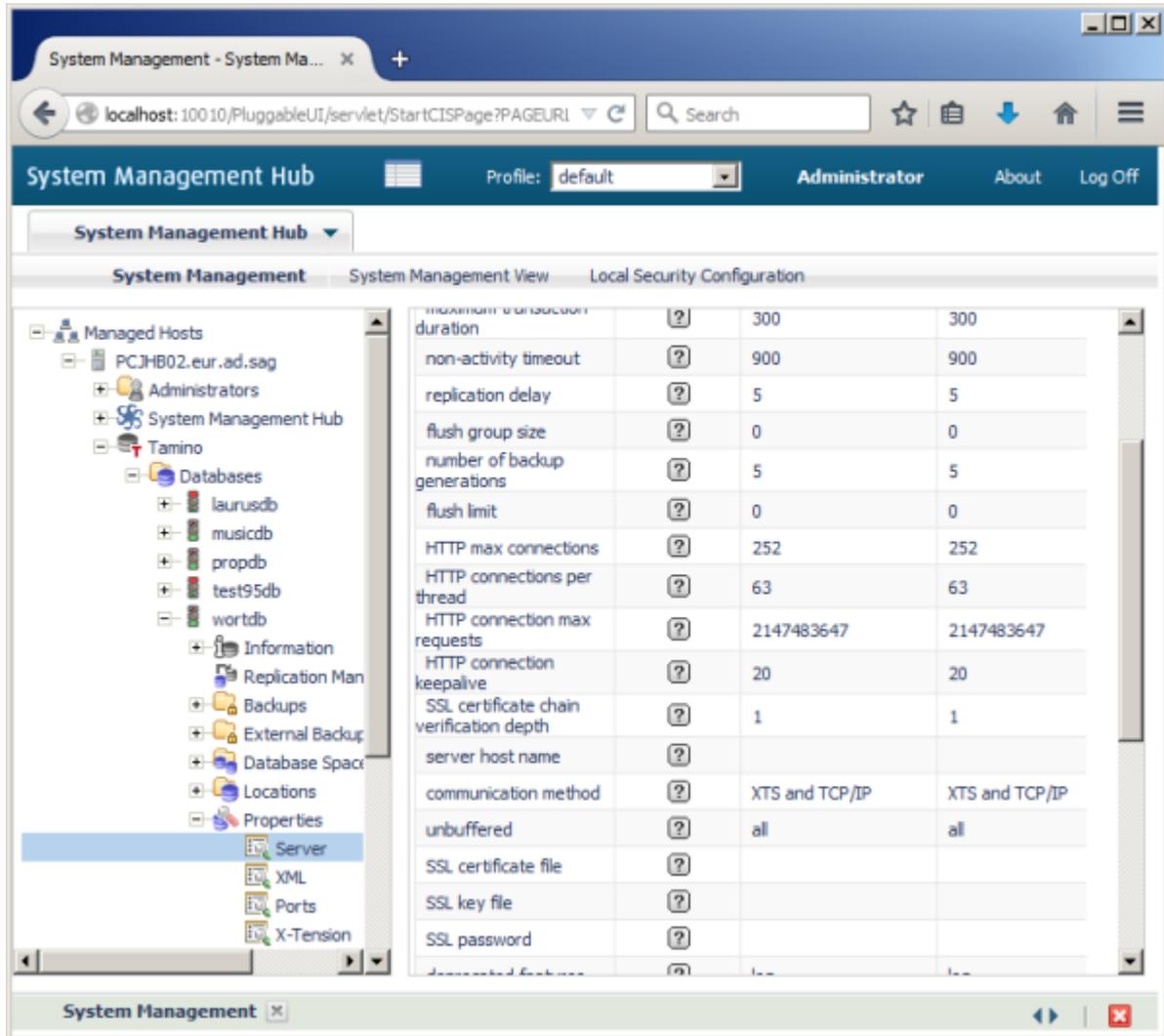


The desired communication methods are specified via the database property `communication method` (see the table “Server properties” in the Tamino Manager documentation, section *Create a Database*).

Communication is stateless: a client issues a request and receives a reply. A *physical* connection between client and X-Machine lasts only for the duration of a request.

## Communication via a Web Server

The communication via web server is the default in that a newly created database specifies 'XTS and TCP/IP' as the communication method database property (see the table “Server properties” in the Tamino Manager documentation, section *Create a Database*).



The screenshot shows the Tamino System Management Hub web interface. The browser address bar displays 'localhost:10010/PluggableUI/servlet/StartCISPage?PAGEURL'. The page title is 'System Management Hub' and the user is logged in as 'Administrator'. The left sidebar shows a tree view of the system management structure, with 'Server' selected under 'Properties'. The main content area displays a table of server properties for a database.

duration	?	300	300
non-activity timeout	?	900	900
replication delay	?	5	5
flush group size	?	0	0
number of backup generations	?	5	5
flush limit	?	0	0
HTTP max connections	?	252	252
HTTP connections per thread	?	63	63
HTTP connection max requests	?	2147483647	2147483647
HTTP connection keepalive	?	20	20
SSL certificate chain verification depth	?	1	1
server host name	?		
communication method	?	XTS and TCP/IP	XTS and TCP/IP
unbuffered	?	all	all
SSL certificate file	?		
SSL key file	?		
SSL password	?		
deprecated features	?	...	...

To communicate with Tamino as such a web server must be configured to redirect the user's HTTP (or HTTPS) calls towards Tamino as described in the section “Configuring the Web Server” in the chapter, section *Before You Start Using Tamino*. Whether the communication between the user and the web server is HTTP or HTTPS depends solely on the configuration of the server and has nothing to do with Tamino communication. Communication from the server towards Tamino is either via TCP/IP or via XTS. XTS-based communication can further be secured using SSL for encryption. The default communication here is XTS. XTS uses TCP/IP also.

In case of TCP/IP communication Tamino listens for incoming requests on the port that is specified via the database property `XML port` (see table “Port properties” in the Tamino Manager documentation, section *Create a Database*). The port number is assigned when the database is created.

In case of XTS communication Tamino listens for incoming requests on the port that is specified via the database property `XML XTS port` (see table “Port properties” in the Tamino Manager documentation, section *Create a Database*). If the property is not specified, the server picks a port when it is started. A typical use case where a port needs to be specified is when client and database are on opposite sides of a firewall, so that a port must be opened there.

A database name is resolved to an IP address and port number via a directory server. The database registers itself with the directory server at startup and deregisters at shutdown. A web server and a database have to use the same directory server to be able to communicate.

### Related Properties

The following database server properties are also relevant for working with XTS (see the table “Server properties” in the Tamino Manager documentation, section *Create a Database*):

Property Name	Meaning
<code>overwrite XTS registration</code>	If a database with the given name is already registered in the directory server, its registry entry is overwritten.
<code>server host name</code>	The hostname that can be optionally entered into the directory server at server startup.

The SSL communication method is identical to XTS, but the communication between the web server and the Tamino database is encrypted. For SSL the following properties are relevant.

Property Name	Meaning
<code>SSL certificate file registration</code>	The absolute path to a file containing an SSL certificate.
<code>SSL key file</code>	The absolute path to a file containing an SSL key.
<code>SSL password</code>	Optionally, an SSL password.

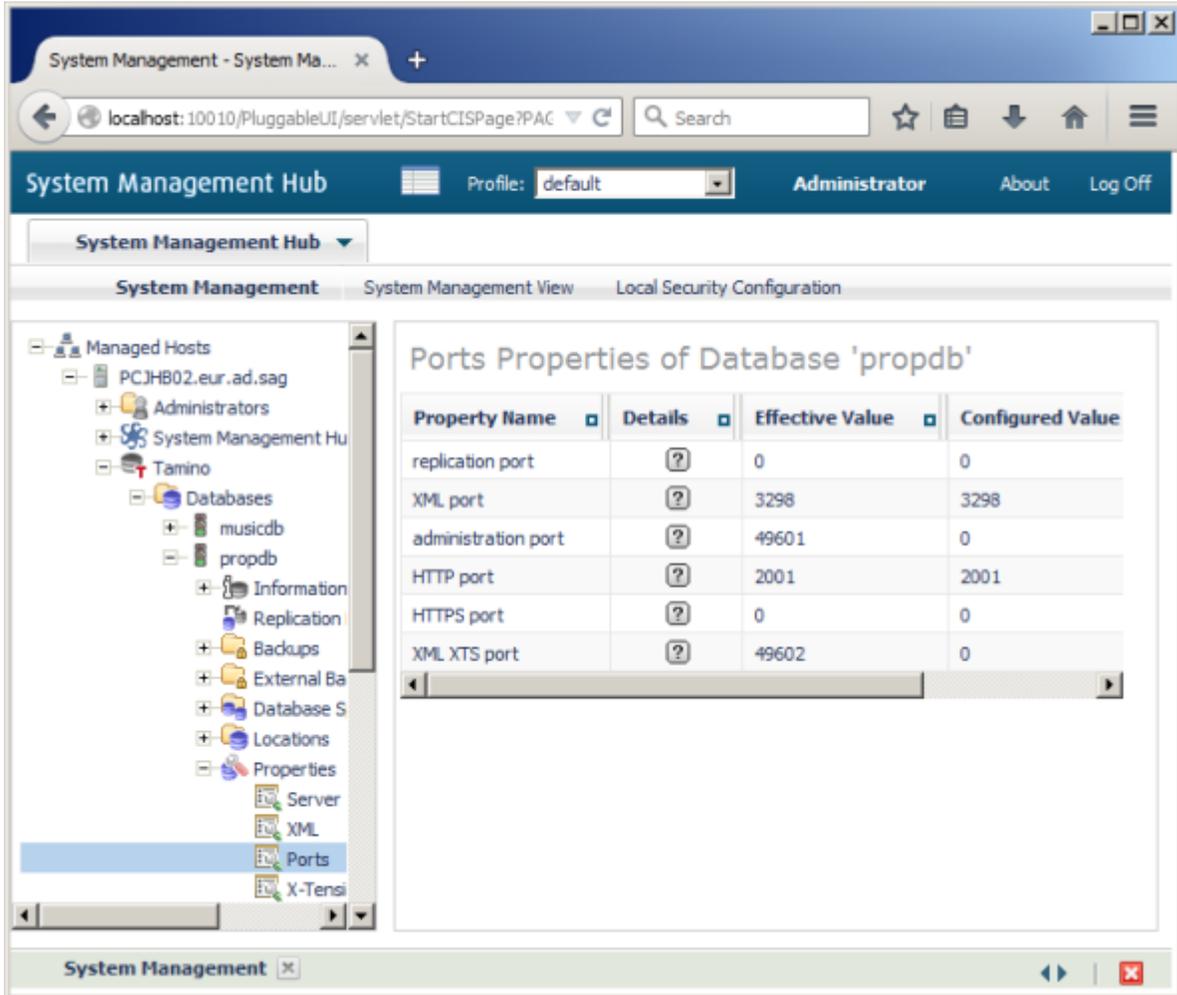
## HTTP/HTTPS based communication

---

Tamino XML Server is able to act as an HTTP server of its own rights. Clients can communicate with Tamino using a native HTTP and/or HTTPS protocol.

This is achieved by configuring a server property communication method (e.g. "HTTP and HTTPS"). A valid port number has to be assigned to the server property `HTTP port`. Via this port, messages based on either HTTP or HTTPS protocol are communicated.

The contents of the HTTP requests and responses in respect to Tamino server request protocol is not affected by this method of communication.



For HTTPS the following properties are relevant.

Property Name	Meaning
SSL certificate file registration	The absolute path to a file containing an SSL certificate.
SSL key file	The absolute path to a file containing an SSL key.
SSL password	Optionally, an SSL password.
SSL certificate chain verification depth	Optionally, the maximum depth for the certificate chain verification.
SSL CA file	Optionally, the absolute path to a CA (Certificate Authority) file.
SSL cipher list	Optionally, a user defined cipher list for OpenSSL communication. Per default the server uses the cipher list "ALL:!ADH:RC4+RSA:!SSLv2:!LOW:!EXPORT:@STRENGTH".

Property Name	Meaning
SSL verify client	Optionally, yes or no.

The Tamino installation provides a self-signed certificate in the directory *\$INODIR/\$IN-OVERS/files/certs* that can be used to test HTTPS communication.

## The "webserviceless" option of the Java or C API

---

The Tamino APIs for Java and for C offer a webserviceless option. Clients can communicate with Tamino without using HTTP. The communication method XTS must be configured in this case. Client and database have to use the same directory server to be able to communicate.

# 3

## Web Servers

---

A client can talk to the X-Machine via HTTP. In the default case, the X-Machine itself does not implement the HTTP protocol. Since high-quality third-party HTTP servers are available, Tamino uses them. Tamino contains a component named X-Port, which is a collection of programs that are installed as interfaces inside web servers. They get control when a request with a URL that is mapped to a Tamino database arrives at the web server. These interface programs forward the request to the target database's X-Machine and send the X-Machine reply back to the client.

The protocol between the client and the web server is HTTP.

Tamino APIs use HTTP to send requests to Tamino via a web server, except of course when using the webserviceless API. Using a web server offers the following advantages:

- No XTS installation/configuration is needed on the client.
- Transparent use of Tamino databases: the client does not need to know whether a requested resource is inside a Tamino database or somewhere else.
- The web server's security infrastructure can be used.
- The web server's logging infrastructure can be used.
- Common firewall setups can be used.
- Browsers can be used as clients

A Tamino database does not have a dedicated web server. A web server can talk to any number of databases. Each database can be accessed by any number of web servers.

Tamino's web server interfaces are compatible with all Tamino versions. It is recommended to use the current version of Tamino's web server interface for communication with all Tamino databases.

The web server does not have to be on the same machine as the Tamino database. It may be desirable for performance or availability reasons to use separate machines for the web server and the database server.

The web server does not have to run on the same platform as the Tamino database. The Tamino distribution only includes web server interfaces for the target platform. Interfaces for other platforms are available; please contact your software supplier for details.

Installing Tamino's web server interfaces is straightforward; all you have to do is to ask the web server to load the interface at startup. Configuration is also straightforward: define a mapping from URLs to Tamino databases, so that the web server knows which requests should be handled by the Tamino interface and the Tamino interface knows to which database it should forward the request. Installation and configuration are usually done automatically by the Tamino installation procedure.

## 4 Webserversless APIs

---

The APIs for Java and C can talk to the X-Machine without HTTP, so no web server is used in this case. The protocol used is the same one that is used between web servers and the X-Machine. From the X-Machine's point of view, such a client is just another web server.



# 5 Security

---

- Introduction ..... 14
- Web Server Assignment ..... 14
- Encryption ..... 14
- Using SSL with Native HTTP access (HTTPS) ..... 15

## Introduction

---

This section deals with security issues of network traffic. Authentication of users and authorization (permissions for parts of documents) are described in the documentation of the Tamino Manager (see the section *Tamino Security*). The X-Machine provides two mechanisms to safeguard communications: it is possible to specify that a given database may only talk to particular web servers, and the communication can be encrypted. The two mechanisms can be combined.

## Web Server Assignment

---

By default, the X-Machine accepts requests from clients without checking their IP addresses. This behavior can be changed by specifying the clients that are allowed to communicate with the database. If at least one client is specified, *all* unspecified clients are rejected. Clients are specified and assigned via the Tamino Manager.

The Tamino Manager uses the term *web server*, but the logic also works for the Tamino Data Loader. Clients are specified in the Tamino Manager with their IP addresses (host/port). The Tamino Data Loader is treated like a web server that runs on port 80.

If a list of web servers is specified for a database, only these clients can access the database.



**Note:** Clients that use the webserversless feature of the APIs cannot access databases that have web servers assigned.

## Encryption

---

The communication between client and Tamino can be encrypted if the XTS communication method is used. The Native TCP/IP communication method currently does not offer encryption. If all communication is to be encrypted, Native TCP/IP must be switched off. A combination of encrypted and unencrypted communication makes sense for those use cases where trusted clients (e.g. web servers that talk to the X-Machine over a secure wire) and untrusted clients should be able to talk to the database. Encryption uses SSL. Please note that the X-Machine currently does not support SSL's authentication capabilities. Clients currently do not use the server certificate to authenticate the server. Encryption is activated via the database property `communication` method. If this property is set to "SSL" (or "SSL and TCP/IP"), then XTS communication is encrypted using the certificate that is specified in the database properties `SSL certificate file` and `SSL key file`. If the used key is password protected, then the password must be specified via the server parameter `SSL password`. A sample certificate is contained in the Tamino distribution (see the directory `files/certs` under the Tamino installation directory).



**Note:** Under Solaris 8, Solaris patch 112438 is required for SSL.

## Using SSL with Native HTTP access (HTTPS)

---

Native HTTP communication with a Tamino server supports the usage of the SSL protocol (TLS 1.0).

The SSL encrypted traffic is handled through the Tamino HTTP port ([link to server properties](#)).

In order to configure the usage of the SSL protocol when talking to a terminal server the following server properties need to be set appropriately:

- `SSL certificate file` contains the full path to the SSL certificate file to be used. Only pem format is supported.
- `SSL key file` contains the full path to the server's private key file
- `SSL password` contains the password in case the various SSL certificate files are password protected
- `SSL CA file` contains the full path to the file containing the CA certificate required
- `SSL verify depth` contains an integer value to indicate the maximum depth of chained certificate verification, the default value is 1.
- `SSL verify client` indicates whether the client is to be authenticated by the server based on its certificate, default value is "no".



# 6 Request Handling in the Server

---

The X-Machine operates in multi-threaded mode. It executes requests in “worker threads”. Incoming requests are queued in front of the worker threads.

A worker thread picks up a request from the queue, reads the input message, executes the contained commands, sends the output message, and then picks up the next request (or waits for one to arrive). The execution of a single request is not multi-threaded: a request stays in its worker thread until it is finished. The number of worker threads determines the number of requests that can be executed in parallel.

A worker thread may have to wait for an event, e.g. I/O completion. If the X-Machine recognizes that all worker threads are waiting, it starts another one. If this happens, a message is written to the job log.

The following database properties are related to request handling:

Property Name	Meaning
XML max simultaneous requests	This specifies the size of the X-Machine's input queue.
XML work threads	This is the initial number of worker threads used by the X-Machine.

Please note that the property `XML maximum sessions` is not related to TCP/IP sessions and has nothing to do with communication resources.



# Index

---

## C

communication methods  
summary, 3

## E

encryption, 14

## R

request handling  
in X-Machine, 17

## T

threads  
summary of worker threads, 17

## W

web servers  
summary of communication with, 9  
webserver  
summary of usage, 5  
webserverless APIs  
summary of communication with, 11  
worker threads, 17

## X

X-Machine  
request handling, 17  
XTS  
related server properties, 6

