



# Frequently Asked Questions About Caché Advanced Security

Version 5.1  
15 June 2006

*Frequently Asked Questions About Caché Advanced Security*

Caché Version 5.1 15 June 2006

Copyright © 2006 InterSystems Corporation.

All rights reserved.

This book was assembled and formatted in Adobe Page Description Format (PDF) using tools and information from the following sources: Sun Microsystems, RenderX, Inc., Adobe Systems, and the World Wide Web Consortium at [www.w3c.org](http://www.w3c.org). The primary document development tools were special-purpose XML-processing applications built by InterSystems using Caché and Java.



The Caché product and its logos are trademarks of InterSystems Corporation.



The Ensemble product and its logos are trademarks of InterSystems Corporation.



The InterSystems name and logo are trademarks of InterSystems Corporation.

This document contains trade secret and confidential information which is the property of InterSystems Corporation, One Memorial Drive, Cambridge, MA 02142, or its affiliates, and is furnished for the sole purpose of the operation and maintenance of the products of InterSystems Corporation. No part of this publication is to be used for any other purpose, and this publication is not to be reproduced, copied, disclosed, transmitted, stored in a retrieval system or translated into any human or computer language, in any form, by any means, in whole or in part, without the express prior written consent of InterSystems Corporation.

The copying, use and disposition of this document and the software programs described herein is prohibited except to the limited extent set forth in the standard software license agreement(s) of InterSystems Corporation covering such programs and related documentation. InterSystems Corporation makes no representations and warranties concerning such software programs other than those set forth in such standard software license agreement(s). In addition, the liability of InterSystems Corporation for any losses or damages relating to or arising out of the use of such software programs is limited in the manner set forth in such standard software license agreement(s).

THE FOREGOING IS A GENERAL SUMMARY OF THE RESTRICTIONS AND LIMITATIONS IMPOSED BY INTERSYSTEMS CORPORATION ON THE USE OF, AND LIABILITY ARISING FROM, ITS COMPUTER SOFTWARE. FOR COMPLETE INFORMATION REFERENCE SHOULD BE MADE TO THE STANDARD SOFTWARE LICENSE AGREEMENT(S) OF INTERSYSTEMS CORPORATION, COPIES OF WHICH WILL BE MADE AVAILABLE UPON REQUEST.

InterSystems Corporation disclaims responsibility for errors which may appear in this document, and it reserves the right, in its sole discretion and without notice, to make substitutions and modifications in the products and practices described in this document.

Caché, InterSystems Caché, Caché SQL, Caché ObjectScript, Caché Object, Ensemble, InterSystems Ensemble, Ensemble Object, and Ensemble Production are trademarks of InterSystems Corporation. All other brand or product names used herein are trademarks or registered trademarks of their respective companies or organizations.

For Support questions about any InterSystems products, contact:

**InterSystems Worldwide Customer Support**

Tel: +1 617 621-0700

Fax: +1 617 374-9391

Email: [support@InterSystems.com](mailto:support@InterSystems.com)

# Table of Contents

- Frequently Asked Questions About Caché Advanced Security..... 1**
  - General ..... 1
  - System Administration ..... 7
  - Auditing ..... 13
  - Programming ..... 15
  - 1 Open Issues ..... 16



# Frequently Asked Questions About Caché Advanced Security

## General

---

### **What is the overall objective of Caché Advanced Security?**

The objective of Caché Advanced Security is simple: to provide a well-designed and well-implemented application with the capability to defend itself successfully against the attacks of malicious users.

### **What is the difference between *authentication* and *authorization*?**

The simplified explanation is:

- Authentication involves verifying your identity.
- Authorization involves checking if you have the permission to do what you want.

During authentication, you undergo what security people refer to as “challenges.” It is very much the way a guard might challenge someone approaching a secure facility:

- Do you have unforged identification documents from a recognized authority?
- Do you know the current password?
- Can you respond to a question with an answer only you are likely to know?

Challenges may even involve collecting and verifying biometric data such as fingerprints, retinal scans, or handwriting samples. Most often, however, authentication of human beings requires that the person know a valid user name and its corresponding password.

Applications, too, may be challenged when they attempt access. Usually they must provide unforgeable certificates of identity before they are allowed to attempt execution of system services.

Once authentication succeeds in identifying users (and applications), the authorization process looks up the following:

- Who you are
- What you are attempting to do
- What data you are attempting to use

If there is a match in the database, the operation proceeds. Otherwise, the system prevents the attempt, usually by signaling some kind of error exception.

### How does Caché Advanced Security work?

In form and function, Caché Advanced Security works like many other computer security systems. It attempts to manage and control the actions that a defined group of subjects are able to take on a known set of protected objects.

In Caché, the subjects are roles or applications; the objects are referred to as resources; and the actions that can be taken are “(R)ead”, “(W)rite”, and “(U)se” defined appropriately for each resource. (Each of these is discussed separately in this document.)

You can visualize the access control as a large matrix with the name of a subjects heading each column and the name of a resource labelling each row. Using this model, each table cell contains the actions that the subject is can perform on the object. For example:

	Analyst	...	Clerk
%DB_CACHESYS	RWU	...	
%DB_DOCBOOK	R	...	
%Service_CSP	U	...	U
%Service_Bindings	U	...	
%Development		...	U
...	...	...	...

When a table cell contains no permission settings, that role has no access to the specified resource.

### Are roles in Caché the same as roles in SQL?

No! The roles defined by Caché and those defined by the SQL standard are *not* the same. They represent two different authorization models that govern two different resources. Caché defines read, write, use, and develop permissions and governs the use of resources defined in the security database. SQL uses select, insert, update, and delete and applies them to its

---

own "resources", namely, users, databases, tables, and so on. Moreover, there are differences between the two approaches that make mapping one into the other problematic. These are in the way permissions are determined and when they are revoked.

They do, however, share the same underlying authentication mechanism so that the same name is used for identifying a user to Caché and SQL.

### **How do roles and users interact?**

Caché Advanced Security does not assign privileges directly to users. Instead, privileges are assigned to roles that can be thought of as position or functional titles within an organization. So, the privileges that Adam Smith possesses are derived from the roles assigned to him, perhaps, the CFO role.

This formulation permits a degree of abstraction, and eases the administrative burden on the System Administrator because the privileges associated with a role usually change less frequently than the people assigned to that role.

### **Why are applications assigned roles?**

Assigning roles to applications allows them to act on behalf of users, possibly with privileges that are different from the user who invokes them. This allows privileges to be better controlled because it permits the exercise of a privilege only under the control of a presumed tested and trusted application.

Further, after checking a user's roles, if the user holds a particular role (called a "match" role), the application can grant the user one or more additional roles (called "target" roles) for the duration of using the application. This process is known as "role escalation."

### **What permissions does Caché Advanced Security use?**

Caché defines the following permissions:

- Read — Permits viewing, but not changing, the contents of a resource.
- Write — Permits viewing and changing the contents of a resource.
- Use — Permits invoking a resource, such as an application or service.

Permissions are often referred to by the first letter of their names; in this case, "R", "W" and "U", respectively.

## What resources are defined under Caché Advanced Security?

Caché comes with the following resources defined by default:

- Database Resources

- %DB\_<DatabaseResourceName>

Examples of databases shipped with the system are %DB\_USER and %DB\_CACHESYS. The predefined resource %DB\_%Default specifies the privileges assumed for a database when:

- It is a database using the (older) 2KB block format.
- No resource name has been assigned to it.
- It is mounted, but is not described in the current configuration.

When mounting an existing database that does not have a database resource name, the default resource, %DB\_%DEFAULT, is assigned to the database.

- Service and System Resources

- %Service\_CSP — Represents the ability to execute a CSP page
- %Service\_CacheDirect — Represents a connection to Caché using Caché Direct
- %Service\_Callin — Represents a connection to Caché via CALLIN
- %Service\_ComPort — Represents a connection to Caché via a Microsoft Windows communications port
- %Service\_Console — Stands for a Caché connections started by the **css** or **csession** command
- %Service\_LAT — Represents a connection to Caché via a Microsoft Windows LAT service
- %Service\_Object — Represents a connection to Caché through one of the available object bindings
- %Service\_SQL — Represents a connection to Caché through ODBC, JDBC or SQL to execute object requests
- %Service\_Telnet — Represents a connection to Caché via the Caché Telnet service for Microsoft Windows
- %Service\_Terminal — Represents a connection to Caché via a terminal on non-Windows systems

- %System\_Callout — Controls calls out of Caché \$ZF(-1), \$ZF(-2), and ! syntax.
- Administrative Resources
  - %Admin\_Manage — Represents the ability to create, modify, and delete Caché configurations, backup definitions, databases, and namespace mappings; and to perform database and journal restores
  - %Admin\_Operate — Encompasses the ability to start and stop Caché, its processes, services, and backups; to mount and dismount databases, and perform integrity checks; to perform database backups; to modify locks; and to examine logs
  - %Admin\_Secure — Represents the ability to administer Caché security
  - %Development — Represents the ability to invoke development tools and capabilities

Resources whose names begin with a percent character (%) are assumed to be Caché defined and managed. Resources defined by the system administrator for a particular site must not begin with this character.

### Why are namespaces not considered resources?

There is no resource which represents any capability on namespaces. The operations which can be performed on a namespace are derived from the databases that are mapped to the namespace.

It is important to understand that using the underlying database privileges can lead to apparently anomalous behavior. Consider a name space, *X*, into which are mapped three databases: *A*, *B*, and *C*. The role *AverageUser* has the following privileges: %DB\_A:R and %DB\_C:R; that is, *AverageUser* has no access to *B*.

In this case, any attempt to access data in *B* fails.

### What are privileges?

Privileges are rights to manipulate a resource. They are formed from a resource name and a permission, for example, %DB\_USER:R. When more than one privilege refers to the same resource, the permissions are often combined: %DB\_USER:RW.

In Caché, privileges are assigned to roles.

### **Why are both users and applications given roles?**

In the Caché model, applications can act as agents for users. To accomplish their tasks, they may need privileges different from the user who invokes them and can therefore have their own roles.

### **How is access to applications controlled?**

Applications are represented by resources. When you make an application known to Caché security, Caché automatically creates an application resource that corresponds to it.

One of the properties of this resource that you can set is called the Protection property. It determines the role a user must have to access the resource, and therefore, execute the application. The possible values of the Protection property are:

- Public — Any user can run the application.
- Restricted — Only users with roles that hold the %Application/<application-name>:Use privilege can run the application.
- Locked — Only users with the %All role are permitted to execute the application.

The Locked setting is intended for emergency use in locking down an application if you suspect that there has been a security breach. Marking an application as Restricted and assigning the roles to users who need to execute the application is sufficient for normal operation.

### **What encryption algorithms does Caché security use?**

Caché implements the Advanced Encryption Standard (AES) from the National Institute for Science and Technology (NIST). The standard is also Federal Information Processing Standard Publication #197 (FIPS PUB 197). It is a fast and strong symmetric algorithm that uses keys of 128, 192 or 256 bits. It can also be freely exported.

The standard is available from the NIST Web site:

- [csrc.nist.gov/publications/fips/fips197/fips-197.pdf](http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf)

The physical implementation is covered by the following copyright:

Copyright (c) 2003, Dr Brian Gladman, Worcester, UK. All rights reserved.

#### LICENSE TERMS

The free distribution and use of this software in both source and binary form is allowed (with or without changes) provided that:

1. distributions of this source code include the above copyright notice, this list of conditions and the following disclaimer;
2. distributions in binary form include the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other associated materials;
3. the copyright holder's name is not used to endorse products built using this software without specific written permission.

ALTERNATIVELY, provided that this notice is retained in full, this product may be distributed under the terms of the GNU General Public License (GPL), in which case the provisions of the GPL apply INSTEAD OF those given above.

#### DISCLAIMER

This software is provided 'as is' with no explicit or implied warranties in respect of its properties, including, but not limited to, correctness and/or fitness for purpose.

## Must I use Caché Advanced Security?

Yes. Security in Caché is always enabled. However, authenticated and authorized administrators can configure it so that the behavior of the system is benign for legacy applications.

## What security certifications does Caché Advanced Security possess?

InterSystems has begun the process to obtain Common Criteria certification for Cache Version 5.1 at level EAL3. More information on the criteria and the requirements for various levels is available at the Common Criteria Web site: [www.commoncriteria.org](http://www.commoncriteria.org).

# System Administration

---

## When do changes in privileges take effect?

Caché maintains a persistent database of the security settings. When Caché starts, it extracts this information and places it into a segment of shared memory that allows quick access to the consolidated settings. While a process is executing, it maintains its own per-process cache of the privileges it has been granted. This is updated as new privileges are needed (and authorized).

Editing roles, privileges, and so on makes changes to the persistent copy of the information. This becomes visible to users or applications the next time they are subsequently authenticated.

### **What authentication mechanisms are available for Caché installations?**

There are four choices for validating the identity of users:

- Kerberos
- Operating-System Login
- Caché Login
- Unauthenticated (no identity validation)

Kerberos provides the most secure means of authentication. The Kerberos Authentication System, developed at MIT, provides mathematically proven strong authentication over an unsecured network. More information on Kerberos is available at [web.mit.edu/kerberos/www/](http://web.mit.edu/kerberos/www/).

Operating-system-based authentication is available for UNIX and OpenVMS. OS-based authentication uses the operating system's user identity to identify the user for Caché purposes. (Kerberos is the underlying Windows authentication mechanism, so there is no OS-based option on that platform.)

The Caché login mechanism maintains a table of hashed password values for each user account; by comparing that value in the table with a hash of the password provided by the user at login, Caché can confirm the user identity.

You can also allow all users to connect without performing any authentication. This option is appropriate for organizations with strongly protected perimeters or in which neither the application nor its data are an attractive target for attackers.

In addition, there are three locations from which the user may request access:

- Local - the user is running on the same system as Caché
- Client - the user application is connecting through a service such as ODBC, or the CSP Gateway
- Proxy - the user is running a CSP application which interacts with Caché on the user's behalf

This gives rise to the following table of options:

	Local	Client	Proxy
Unauthenticated	Yes	Yes	Yes
Caché Login	Yes	Yes	Yes
Operating-System Login	Yes	No	No
Kerberos	Yes, but ...	Yes, but ...	Yes, but ...

The reason for the “Yes, but ...” for the Kerberos entries is that Kerberos offers a number of strategies for validating identity. However, the options actually available depend on the details of the location and the services which will be used. A detailed discussion is beyond the scope of this document.

### What roles are predefined in Caché?

Caché has five roles built in:

- %All
- %Developer
- %Manager
- %Operator
- %SQL

The default privileges associated with each role are given in the following table and discussion:

	%Developer	%Manager	%Operator	%SQL
%Admin_Manage		U		
%Admin_Operate		U	U	
%Admin_Secure		U		
%DB_CACHEAUDIT/<Role>		R		
%DB_CACHELIB/<Role>	R	RW		
%DB_CACHESYS		RU		
%DB_CACHETEMP	RW	RW	RW	
%DB_DOCBOOK	R	RW	R	
%DB_SAMPLES	RW	RW		

	<b>%Developer</b>	<b>%Manager</b>	<b>%Operator</b>	<b>%SQL</b>
%DB_USER	RW	RW		
%DB_%Default	RW	RW		
%Development	U	U		
%Service_CacheDirect				
%Service_Callin				
%Service_ComPort				
%Service_Console	U	U		
%Service_CSP	U	U	U	
%Service_LAT				
%Service_Object	U	U		
%Service_Bindings	U	U		U
%Service_Telnet	U	U		
%Service_Terminal	U	U		
%Service_DCP				
%Service_ECP				
%Service_Monitor				
%Service_MSMAActivate				
%Service_Shadow				
%Service_WebLink				
%Service_WebService				

Except for %All, the use of these predefined roles is optional. This role, however, is treated specially:

- The %All role cannot be removed from the system.
- The privileges of %All cannot be modified.
- %All is automatically granted privileges on any new resource defined in the system.
- %All must always have at least one user assigned to it. Caché prevents any attempt
  - to invalidate the last user from %All, or
  - to remove a user if that user is the only one assigned to the %All role.

## Can I mistakenly configure Caché security so everyone is locked out?

As noted previously, Caché has special checks to assure that there is at least one valid user assigned to the %All role, and that the %All role contains all permissions on all resources. Thus, there is at least one user that has sufficient access to administer the system.

It should be noted, however, that this is not a panacea. Consider the case that all other users have been deleted. This lone user must still be authenticated to Caché. If authentication requires a password, and no one can supply it, then no user can access Caché.

## What do the terms “Public”, “Restricted”, and “Locked” mean when referring to applications?

Applications in Caché are classified in three groups:

- Those labeled “Public” can be executed by any user.
- User attempting to execute an application in the “Restricted” category must be assigned to a role that grants them “Use” permission on the application.
- “Locked” applications can only be executed by a user who is assigned to the %All role.

## What do I need to be aware of when installing or upgrading to Caché Advanced Security?

An installation of Caché Advanced Security will start up in slightly differently from prior versions of Caché. This is true for both new installation and upgrades from prior versions. System administrators should be aware of the following differences after installation:

- After installation, the system security parameter, *PercentGlobalWrite*, is set to the value `RESTRICTED`.

In Caché, routine and global names that begin with a percent-sign (%) are handled specially; they are visible in all namespaces, not just in the namespace containing the database where they are stored. The *PercentGlobalWrite* parameter governs access to these “percent” globals by non-percent routines stored in other databases.

A value of `RESTRICTED` means the modification of percent globals is subject to normal security control. If *PercentGlobalWrite* is set to `PUBLIC`, all users have implicit Write permission for percent globals (the default behavior in previous versions).

- All users require new passwords assigned to them after an upgrade installation.

The password hash function used in Caché Advanced Security is more robust than those used in earlier versions of Caché. Since Caché only stored (and stores) the hashed form of the password for comparison there is no simple way to invert the hashed form (giving a plaintext password) and replace it with the hashed value using the newer function. As a result, to take advantage of this robustness, users need to enter new passwords.

- The *DefaultSecurityDomain* parameter value is initialized to be the machine name Caché is installed on.

Whenever a user identifier without a domain name is used, the default domain is assumed. In other words, if the default domain is *InterSystems.com*, the user identifiers *Paul* and *Paul@InterSystems.com* are equivalent. In a single domain configuration, use of a domain name other than the default domain is prohibited.

Visibility of domain names during normal use is controlled by the *SecurityDomains* configuration parameter. If the value of this parameter is `SINGLE`, then `$USERNAME` does not include the domain name, and system utilities do not show the domain name when displaying user names. If the value of this parameter is `MULTIPLE`, then `$USERNAME` includes the domain name, and system utilities show the domain name when displaying user names.

- By default, developers do not have privileges on many of the Caché services they did under prior versions.

The default installation of Caché is conservatively configured. The predefined roles do not include privileges for legacy resources such as COM ports or LAT, which most customers do not need. As necessary, administrators can alter the predefined roles or create new roles that require a different set of privileges to meet the needs of each site.

### **What other operational aspects of Caché Advanced Security are different from previous versions of Caché?**

Apart from the obvious constraints established by the security settings defined by the system administrator, an installation of Caché Advanced Security operates slightly differently from prior versions of Caché. This is true for both new installation and upgrades from prior versions. For example:

- For CSP applications, security information is maintained as part of the CSP session. That is, the values of `$USERNAME` and `$ROLES` are preserved across page requests, even if different processes are used to execute those requests. (More specifically, when processing begins for a CSP page, `$ROLES` contains the user's roles as well as roles defined for the application.

It does not contain roles that were dynamically added during processing of a previous page via SET \$ROLES or \$SYSTEM.Security.AddRoles( ). This is true for both “stateless” and “state-full” sessions.

- Recompilation of a privileged routine automatically voids any privileges it has.

After recompilation, an authorized user with the proper privileges must reestablished privileges for the recompiled routine.

- Caché ObjectScript now has two new special system variables, *\$USERNAME* and *\$ROLES* that help applications use Caché Advanced Security to manage their own security needs. The meaning of these variables is described in the [Programming](#) section of this document.

### Can Caché encrypt its databases?

Yes, the administrator may set up the system so that one or more of the databases (cache.dat files) Caché uses are encrypted.

Database encryption in Caché functions entirely within the Caché application. It is entirely separate from any host operating-system facilities provided for encryption of data such as the Windows Encrypted File System (EFS) or Loop-AES facility available for use with Red Hat Linux. Caché provides key management, encryption and decryption operations.

For details, see the “[Database Encryption](#)” chapter of the *Caché Security Administration Guide*.

## Auditing

### Does Caché Advanced Security provide an audit trail?

Yes, Caché can record events of note in a tamper-resistant log. It also provides the same capability to applications; they can record application-defined events in the same log.

Event auditing can be turned on or off as a whole by the system administrator. When auditing is enabled, some system events are considered mandatory and are always recorded. For all other events, finer control over whether an entry is placed in the audit log can be exercised.

### What are the predefined Caché audit events?

The events predefined by Caché are given in this table:

Type	Event	Represents	Mandatory?
%System	Start	Caché starts	Yes
%System	ConfigurationChange	Caché successfully starts with a configuration different than the previous start, or new configuration is activated while Caché is running	Yes
%System	Stop	Caché is shut down	Yes
%Login	Login	Successful login	No
%Login	LoginFailure	Unsuccessful login attempt	No
%Login	Logout	A user leaving the system	No
%Security	UserChange	Definition of a user created, changed, or deleted	Yes
%Security	ApplicationChange	Definition of an application created, changed, or deleted	Yes
%Security	RoleChange	Definition of a role created, changed, or deleted	Yes
%Security	AuditChange	Audit is stopped or started, entries are erased or deleted, or the list of events being audited is changed	Yes
%Security	ServiceChange	Security settings for a service changed	Yes
%Security	SystemChange	System security settings changed	Yes
%Security	ResourceChange	Definition of a resource created, changed, or deleted	Yes
%Security	DomainChange	Definition of a domain created, changed, or deleted	Yes
%Security	Protect	A security protection error is given to a process	No
%Security	AuditReport	Any standard audit report is run	Yes

Type	Event	Represents	Mandatory?
%DirectMode	DirectMode	Any command is executed in direct mode	No

In addition, any number of application-defined events may also be recorded in the audit log. The class, %SYSTEM.Security.Audit, is provided for this purpose.

### What is recorded in the audit log?

The audit log always contains the date and time of the event, the event's name and type, the application or system function that reported it, the user on whose behalf the activity was running. The event also contains other descriptive information related to the particular event.

## Programming

### What is the purpose of \$USERNAME?

*\$USERNAME* is a special variable added to ObjectScript as part of Caché Advanced Security. *\$USERNAME* is a read-only variable. It contains a string giving the authenticated name of the current user.

### What is the purpose of \$ROLES?

Like *\$USERNAME*, *\$ROLES* is a special variable connected with security. *\$ROLES* contains a comma-separated list of the role names assigned to the current user. The union of all the privileges granted to all roles in the list determines the privileges the user possesses.

*\$ROLES* initially contains the roles assigned to the current user when he or she is authenticated. This set of role names is referred to as the “login roles” .

Applications can affect user privileges by modifying the contents of *\$ROLES* by appending role names to the list. The roles which apply at any instant in the execution of an application are called the “active roles” .

To assist in changing the active roles, the **NEW** command treats the variable, *\$ROLES*, specially. Rather than leave *\$ROLES* empty, the **NEW** command copies the previous contents of *\$ROLES* to the newly created instance. (This is what the developer desires in most cases, and is easily undone if not.) Now the application is free to modify the list and, whether control leaves the containing block normally or abnormally, the changes are undone upon exit.

One fact must be remembered, however. When determining the active roles, the names used are a union of those presently in *\$ROLES* together with those that formed the login roles. Thus, it is not possible to execute with a set of privileges more restrictive than one has at the time of authentication.

### **What is the value of *\$USERNAME* if no authentication is being done?**

If no authentication of users is being done, the value of *\$USERNAME* depends on the service the user is attempting to run. Each service has a default value for *\$USERNAME* and this value is set before the service begins to execute. The default value cannot be an empty string, but otherwise it can be set by the system administrator.

## **1 Open Issues**