

# Red Hat Identity Management and Security Solutions

By Sean Cotter  
Product Manager, Red Hat Directory and Security Products

## Abstract

Red Hat identity management and security solutions are designed to:

- Simplify management of identity information in complex heterogeneous environments.
- Bring the benefits of the open source development model to enterprise identity and security infrastructures.
- Provide a clear path to integrated identity and security control as processes and solutions continue to evolve.

This paper describes how Red Hat's acquisition and development of Netscape directory and security solutions lay the foundation for meeting these goals.

Revision 1.0 – September 26, 2005

## Table of contents

Meeting the challenge	2
Support for Identity Management processes	5
Red Hat Directory Server:	
Deployment scenarios	9
Technical highlights	13
Red Hat Certificate System	
Enrollment scenarios	17
Technical highlights	24

## Meeting The challenge

The challenges of identity management and security are familiar to organizations of all types and sizes:

- User problems with passwords, access, privacy, and security disrupt productivity and continue to account for a large percentage of IT costs.
- Extending access to more users with vastly different needs and roles requires more flexible systems for managing user information while reducing per-user administrative costs.
- Poor scalability, reliability, and interoperability of user management systems in heterogeneous environments lead to unpredictable delays in deploying mission-critical applications.
- Expenses associated with increasingly complex auditing and compliance requirements multiply when user data and access rights can't be retrieved or modified rapidly and accurately.

In the open source ecosystem that Red Hat is leading, an identity management solution must:

- Simplify management of increasingly complex environments.
- Provide the benefits of the open source development model to all components of an enterprise deployment.
- Provide a clear path to integrated identity and security control as processes, systems, and third-party solutions evolve.

## Red Hat and Netscape

Red Hat acquired key Netscape Security Services technologies in December 2004 and released Red Hat® Directory Server and Red Hat Certificate System in June 2005. As a result, Red Hat can now offer customers mature, widely deployed enterprise products that lay the foundation for a complete identity management solution.

Reborn as Red Hat Directory and Security products, the Netscape technologies and engineering team acquired by Red Hat have a collective history going back to the late 1990s, when Netscape engineers developed LDAP, SSL, and other

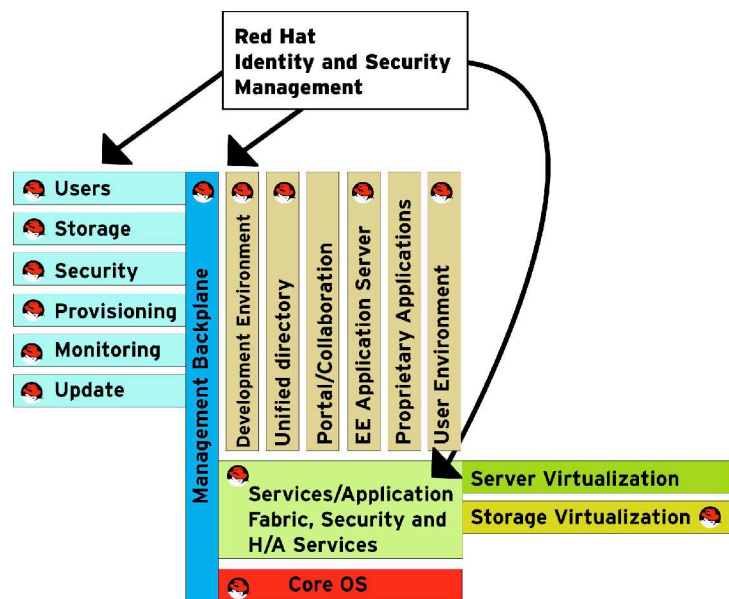
fundamental Internet technologies and built them into groundbreaking server products. The track record they bring to the Red Hat open source architecture includes:

- Years of experience with large deployments to hundreds of organizations and millions of users across government, education, and industry.
- Releasing the Network Security Services (NSS) crypto engine and related Netscape technologies into the open source community as Mozilla projects.
- Periodic revalidation, with Sun Microsystems, of NSS under FIPS 140-1, a requirement for all government deployments that involve cryptographic software.
- Periodic revalidation of Red Hat Certificate System under Common Criteria, also a government requirement.

### Red Hat identity and security management

In the long term, Red Hat Directory and Security Products are positioned to provide the following key benefits for all Red Hat customers:

- Leveraged value throughout the open source infrastructure
- Ecosystem participation from ISVs, IHVs, and the community
- Increased adoption through standards-based development and extended features



With the release of Red Hat Directory Server and Red Hat Certificate System in June 2005, Red Hat began to build out the identity and security elements of the Open Source Architecture.

These products are summarized briefly below. The rest of this document outlines some typical deployment scenarios and provides more details on key features.

**Red Hat Directory Server:  
scalable identity**

Red Hat Directory Server (formerly Netscape Directory Server) provides a mature, highly scalable and reliable hierarchical data store designed to answer two questions about a user attempting to gain access to any system:

- *Authentication: Who are you?*
- *Authorization: What can you do?*

Red Hat Directory Server enables appropriate and timely authentication and access to resources and applications for all users in an organization, using a standards-based, highly scalable, high-performance server architecture.

In addition, Red Hat Directory Server:

- Improves user productivity by supporting single-sign on and simplified access control.
- Centralizes user data and simplifies management, thus reducing costs.
- Provides a rich set of tools, SDKs, and APIs to support rapid development.
- Facilitates compliance with Sarbanes-Oxley (SOX), Federal Information Processing Standard (FIPS) 201, and other regulatory directives.

Red Hat released the core Directory Server code to the open source community via the Fedora Directory Server project in June 2005. The remainder of the server code, including management software, will be released by the end of 2005.

## **Red Hat Certificate System: simplified assurance**

Red Hat Certificate System (formerly Netscape Certificate Management System) provides a highly scalable, easily managed end-to-end digital credential solution that's tuned to answer the third key identity management question for many highly secure business and government organizations.

- *Assurance: Are you who you say you are?*

By dramatically simplifying the end-user experience of enrollment, the smartcard management system that's built into Red Hat Certificate System can bring costs of middle- and high-assurance smartcard solutions within reach for organizations that previously couldn't afford them.

In addition, Red Hat Certificate System:

- Supports other standards-based identity management systems for organizations with medium- to high-security needs.
- Lays the groundwork for HSPD #12/FIPS 201 compliance.
- Provides high scalability proven in very large deployments.
- Is undergoing FIPS 140-2 and Common Criteria recertification for core components to support government deployments.

## Support for identity management processes

Identity management involves several distinct processes:

- *Registration and provisioning* typically happens once.
- *Issuance of credentials* typically happens periodically.
- *Life cycle management* of the identities and credentials occurs continuously until the user is deprovisioned.

### Registration and Provisioning

The first stage of identity management establishes an identity by collecting and verifying information about a new user.

Today, Red Hat Directory and Security products leverage open standards to support many forms of integration with new or existing systems and processes, including:

- *Customization* of enrollment processes helps organizations to enforce appropriate policies for varying types of users and procedures.
- *Synchronization and integration* of user data from other directories and databases ensures that user information acquired during registration is propagated correctly to all relevant systems. Windows Sync integration is available out of the box. Requirements for synchronizing with databases such as Oracle or PeopleSoft tend to differ for each deployment and typically involve writing scripts using PerLDAP or similar tools. Sample scripts are available as part of the LDAP SDK.
- *User provisioning* provides mechanisms for adding new users to a corporate directory (for example, as the result of a face-to-face interview or through custom database synchronization scripts) and assigning them appropriate access rights.
- *Logging and auditing* permits verification of evidence and decisions involved in the registration process after the fact.

Upcoming releases will support additional forms of integration with third-party software under FIPS 201 and related standards:

- Integration with workflow software to support customizable, verifiable registration processes.
- Integration with biometrics devices to support mechanisms for collecting, storing, and using biometric data.

### **Credential issuance**

Newly registered users need credentials such as passwords, Kerberos authentication, certificates, or access badges to gain access to:

- Physical locations such as labs or offices
  - Online access to resources such as servers and applications
- Today, Red Hat Directory and Security products support the issuance, integration, and management of a wide range of credentials, including:

- User names and passwords
- Kerberos authentication via SASL GSS/API
- Certificates (PKI credentials)
- Smartcards

Red Hat Certificate System and Red Hat Directory Server provide a fully integrated PKI and end-to-end smartcard management and directory solution that **Personal Identity Verification (PIV) smartcards** will be able to plug into easily.

### **Life cycle management**

Today, Red Hat Directory and Security products support key technologies required to manage identities and their credentials from the time a user first registers throughout the lifetime of that user within an organization. These include:

- Lightweight Directory Access Protocol (LDAP):
  - Authentication and access control, including support for NIS, PAM, SSH, and other system applications
  - Self-service portals

- Certificate verification:
  - Certificate Revocation Lists (CRLs)
  - Online Certificate Status Protocol (OCSP) Responder
- Integration with mail, web, and other LDAP-enabled applications:
  - Typedown addressing for clients such as Outlook and Thunderbird
  - Preference storage for servers
  - Authentication for websites
- *Synchronization* with user data stores (such as Windows Active Directory) ensures that changes to user data in one location are propagated correctly to other locations.
- *Red Hat Management Console* provides a consistent GUI to administer both Red Hat Directory Server and Red Hat Certificate System server architectures:
  - Server functions can be administered remotely
  - Change configurations, data, schema without downtime
  - Access server configuration, monitoring info via LDAP
- **Web applications** make it easy to provide web access to directory and certificate information via phonebook, org chart, and mapping interfaces.

Upcoming Red Hat Directory and Security product releases will build on this foundation to support improved self-service, password, policy management, and auditing features.

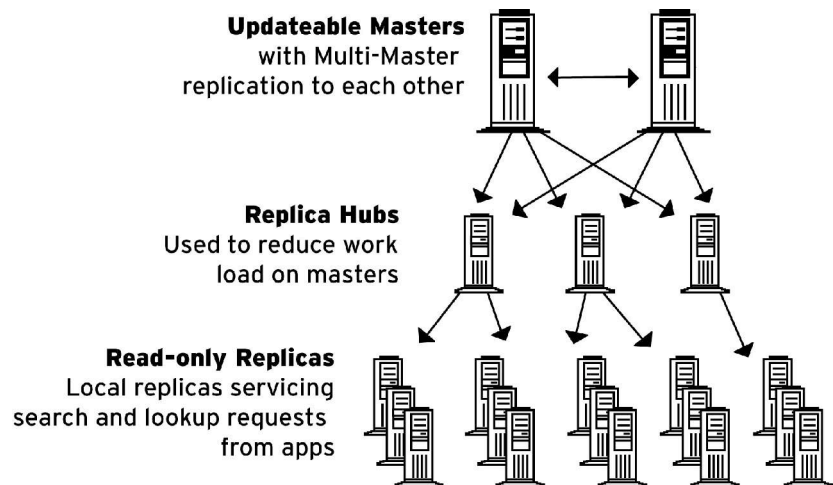
## Red Hat Directory Server: deployment scenarios

Red Hat Directory Server has been designed and tested for nearly a decade to support complex, globally distributed directory deployments with high-availability requirements.

High-availability directory deployments typically use multimaster replication to provide the following strategic benefits:

- Master copies reside on multiple servers.
- Masters can be situated in different data centers, different geographic areas.
- Changes to data can be made to closest server, and are then propagated to the other masters.
- Failover ensures continuous service.
- Automatic time-based conflict resolution simplifies administration.

These are the basic building blocks of multimaster replication:



The following scenarios provide simplified examples of the use of multimaster replication in distributed directory deployments.

### Scenario: Enterprise application with high search and update rates

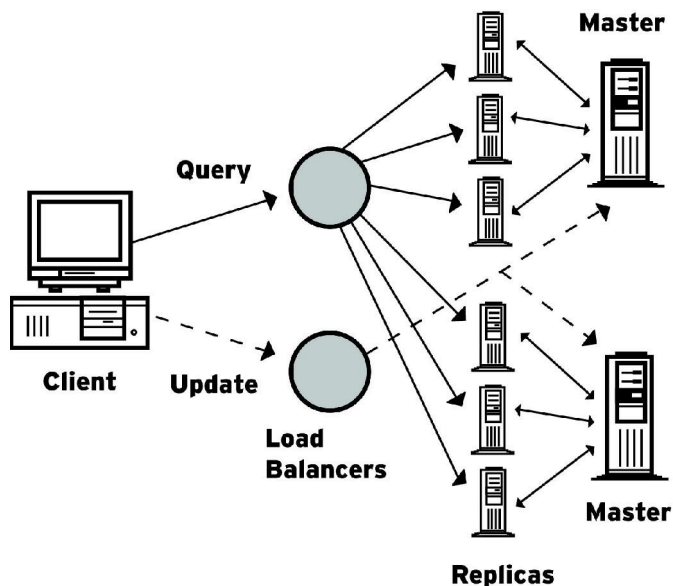
Many large e-commerce sites use LDAP directories to service a high rate of both customer queries and changes to customer-specific information.

The diagram below shows two masters and multiple read-only replicas. Updates and search requests are round-robin as follows:

- A query load balancer routes search requests to read-only replicas.
- An update load balancer (hardware or software) routes update requests directly to one of the masters.

The load balancers can be implemented in hardware or software, separately or as a single multi-purpose node.

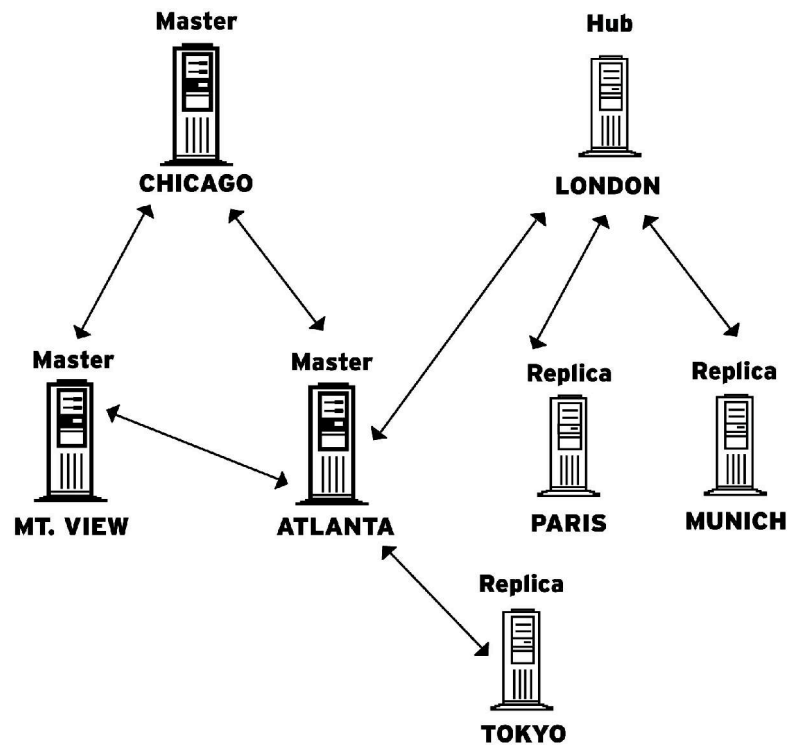
In this scenario, masters can concentrate on rapid responses to update requests and replicating them to other servers, while replicas handle more frequent search requests.



### Scenario: Use of hubs and replicas in a global network

Global organizations service geographically distributed offices. Depending on what kinds of business the organization does in which areas, local directory needs and architectures may vary significantly.

To supply directory services as efficiently as possible, masters and hubs can be arranged to account for distributed query patterns as well as failover and disaster recovery.



In the simplified scenario illustrated above, three master servers service the three regions where the largest numbers of employees are concentrated. These masters integrate company-wide information from a variety of databases and other sources for use by authenticated users and administrators worldwide. Such masters might well have their own associated replicas to handle the local query load, but that level of detail is not shown.

The European offices are also located in three main centers. A hub directory in London services replicas in Munich and Paris. Local queries get routed to local replicas, while less frequent

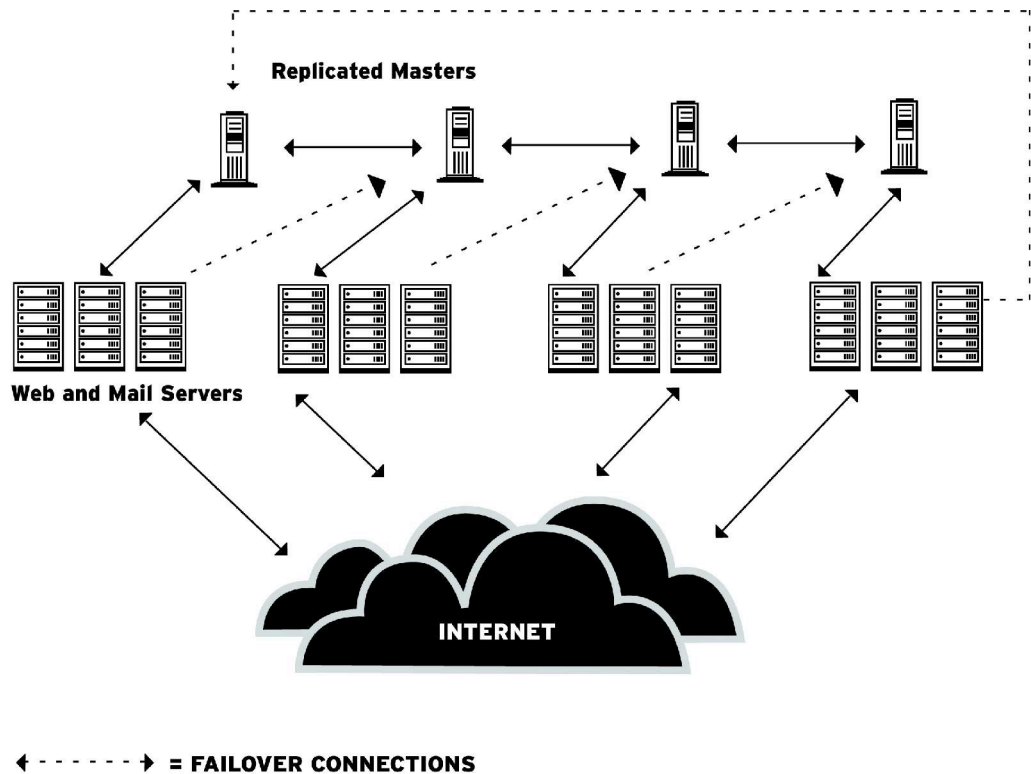
update requests are routed to the US masters for propagation throughout the global infrastructure.

The Tokyo office has no need for a hub. Its directory communicates directly with the master in Atlanta.

### Scenario: Multimaster replication and failover with regional servers

Multimaster replication is commonly used as part of a failover strategy. In the example below, geographically distributed web and mail servers serve local clients and normally communicate with their closest master.

If a master fails, the servers it normally serves automatically switch to the master in the next region. In this manner, geographic load balancing can be combined with failover and disaster recovery.



## Red Hat Directory Server: Technical highlights

### Authentication

Red Hat Directory Server supports a variety of authentication standards and technologies, including:

- Username and password, Digest MD5
- OS logins via NIS or PAM
- Kerberos tickets via SASL/GSSAPI
- PKI authentication based on TLS (SSL) and X.509 certificates
- Impersonation (proxy) for multi-tier client applications

In addition, customers can write their own plug-ins using a robust and well-documented plug-in API for virtually any legacy system or database applications, such as Oracle.

Built-in password management capabilities include:

- Identity synchronization with Windows Active Directory
- Hack protection—login tries, account lockout
- Crack protection—minimum length, no "trivial" passwords, password history
- Selectable password storage (clear, crypt, SHA1)

### Authorization/access control

Red Hat Directory Server also an extremely flexible, easily configurable, fine-grained access control mechanism. Access can be controlled based on external criteria or on groups and roles, and customized in a hierarchical fashion.

Access can be set for an entire Directory Information Tree (DIT) , for subtrees, or individual entries, and can differ for individual attributes within an entry.

For example, password policy can be set across an entire DIT, then overridden for subtrees or even individual users based on specialized needs. In this way, passwords for highly sensitive applications can be required to meet more rigorous guidelines, get changed more frequently, and so on.

Red Hat Directory Server can also be used to store access control information for devices, servers, and other resources. Access controls can be based on user or group membership, IP or domain name, time of day, and many other criteria.

### **Flexible administration**

It's rarely necessary to restart the server. Most configuration changes, importing and exporting, schema changes, and indexing can be done on the fly without downtime.

Configuration settings are stored in a special `c=config` subtree, which means they can be accessed or searched by means of standard LDAP queries.

Monitoring information is stored in a `cn=monitor` subtree, so that status or statistics on things like the number of threads in use at any given time can also be retrieved via standard LDAP queries.

SNMP support is also built into the product, so information on variables such as disk space can be monitored remotely.

As discussed above, most common administrative tasks can be performed from the Red Hat Management Console, a Java application that allows administrators to perform administrative tasks remotely. Administrators can also use configuration files or command line utilities for remote administration.

### **Groups, roles and class of service**

A *static group* consists of a single entry that contains a list of member entries. A *dynamic group* allows you to filter entries that contain a particular attribute and include them in a single group.

A directory tree organizes information hierarchically. This hierarchy is itself a grouping mechanism, though it is not suited for changing organizations.

*Roles* unify static and dynamic groups. Each entry assigned to a role contains a computed attribute that specifies all of the roles an entry belongs to. A client application can check role membership by searching the attribute, which is computed by the directory and therefore always up-to-date.

Roles are designed to be more efficient and easier to use for applications. For example, applications can locate the roles of an entry rather than select a group and browse the members list.

*Class of service (CoS)* allows you to share attributes between entries in a way that is invisible to applications. With CoS, some attribute values may not be stored with the entry itself. Instead, they are generated by class of service logic as the entry is sent to the client application.

For example, a directory typically contains thousands of entries that all share the common attribute `facsimileTelephoneNumber`. Traditionally, to change the fax number, you would need to update each entry individually, a large job for administrators that runs the risk of missing some. With CoS, you can generate the attribute dynamically. The `facsimileTelephoneNumber` attribute is stored in one place, and each entry points to that place to give a value to their fax number attribute. For the application, these attributes appear just like all other attributes despite not actually being stored on the entries themselves.

### **Reliability and scalability**

To ensure reliability, scalability, and geographic distribution, Red Hat Directory Server supports four-way multimaster replication, and chaining.

Support for multimaster replication is discussed above under Directory Server Scenarios, and includes support over WANs with poor or intermittent connectivity.

Chaining refers to the ability to create a special entry in a subtree of a directory. All LDAP operations attempted below this entry are sent to a remote machine where the entry is actually stored. Chaining can be used both for load-balancing and to permit the geographic distribution of data so that it remains physically close to the resources need access most frequently.

## Tools and SDKs

The following Mozilla SDKs are used extensively in development:

- Open source Mozilla LDAP C SDK (includes command line utilities, e.g. ldapsearch)
- Mozilla LDAP Java SDK
- Standard Perl tools for LDAP just work
  - PerLDAP (<http://www.perldap.org/>)
  - Perl-LDAP (<http://ldap.perl.org/>)

Additional SDKs for LDAP products will also work fine, including:

- Java Naming and Directory Infrastructure (JNDI)
- PHP LDAP
- Python LDAP

Other tools include support for:

- Load and performance testing
- Access log analysis
- Ethereal (open source network sniffer)

## Plug-in API

Many customizations, as well as standard features, use the Plug-In API (C/C++). Plug-ins can be used to:

- Design an action that the Directory Server performs before the server processes an LDAP action (Filters). For example, you can write a custom function to validate data before the server performs an LDAP operation on the data.
- Design an action that the Directory Server performs after the server successfully completes an LDAP operation (Triggers). For example, you can send mail to a client after an LDAP operation is successfully completed.
- Define extended operations as defined in the the LDAP v3 protocol.
- Provide alternate matching rules when comparing certain attribute values.

## Red Hat Certificate System: Enrollment scenarios

Red Hat Certificate System provides a fully integrated, end-to-end solution for deploying and managing digital IDs, including:

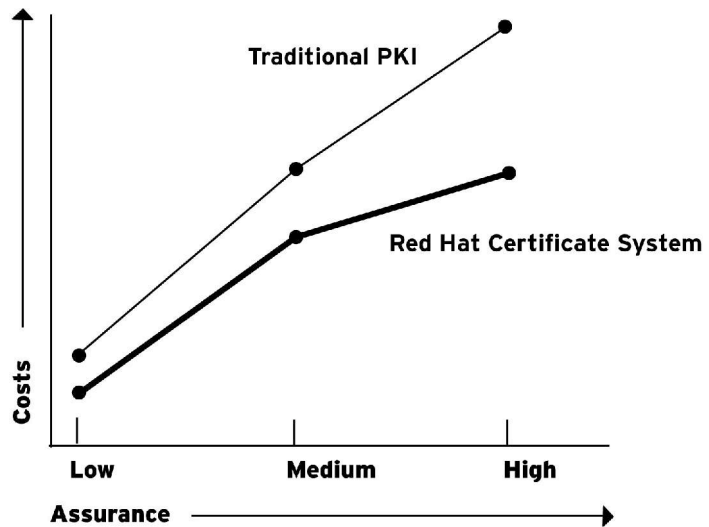
- Certificate Authority (CA)
- Smartcard management system/registration authority
- Private key archival system
- Online Certificate Status Protocol (OSCP) Responder
- Client middleware

Starting several years ago at AOL/Netscape and continuing with the Red Hat acquisition, the Directory and Security Products team has focused usability improvements for end users and administrators as well as overall integration.

As a result of these efforts, Red Hat Certificate System dramatically lowers PKI deployment and management costs:

1. **Lower development costs.** All elements of a complete PKI solution are all developed together by a single team.
2. **Lower software costs.** Because all elements of a complete solution are part of a single Red Hat subscription, software typically costs less than comparable software purchased separately.
3. **Lower support and administration costs.** Usability breakthroughs provided by integrated smartcard management benefit both end users and administrators.

In short, Red Hat Certificate System reduces the costs that until now have plagued different levels of PKI assurance:



Red Hat Certificate System uses Red Hat Directory Server as the internal certificate database, leveraging its replication, chaining, and failover capabilities to support CA cloning and geographic distribution.

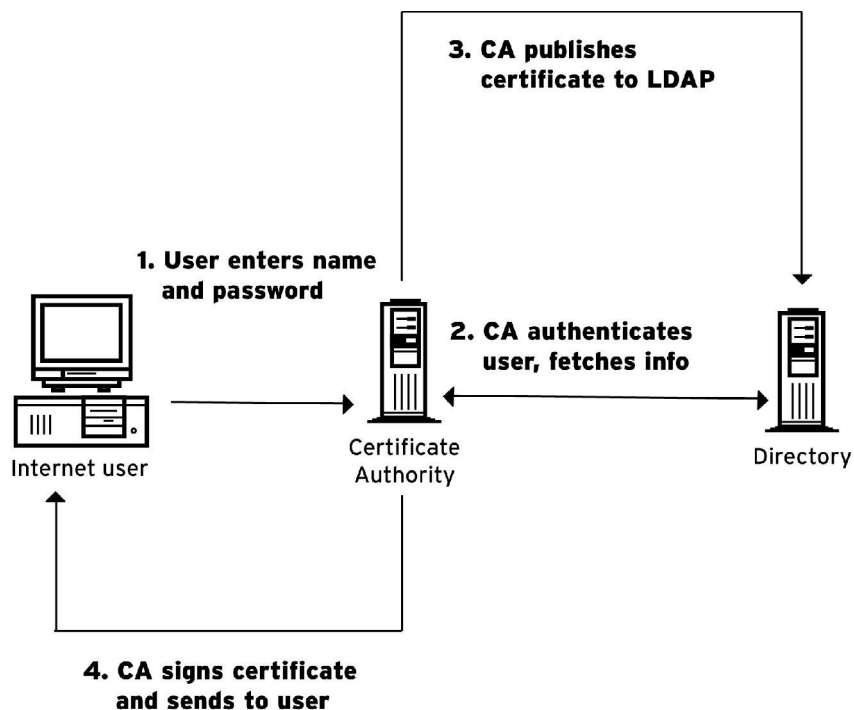
Red Hat Certificate System also includes the **Enterprise Security Client**, a small client application available for Red Hat Enterprise Linux, Windows, and Mac OS X that vastly simplifies the user experience of smartcard enrollment.

Whether used to support software certificates, smartcards, or specialized tokens such as Personal Identity Verification (PIV) cards specified for use by Federal agencies under FIPS 201, Red Hat Certificate System supports a wide range of enrollment scenarios and life cycle management requirements.

The enrollment scenarios that follow describe three deployments that meet low, medium, and high-assurance requirements and exemplify the Red Hat approach to PKI.

## Enrollment Scenario: Customer identities, low assurance

The first scenario shows a customer using a web interface to request a certificate using a name and password. The request includes a public key whose corresponding private key remains on the user's computer.



When the CA receives the request, it checks the name and password against the Directory Server, processes the request, and formulates a certificate for use in SSL client authentication.

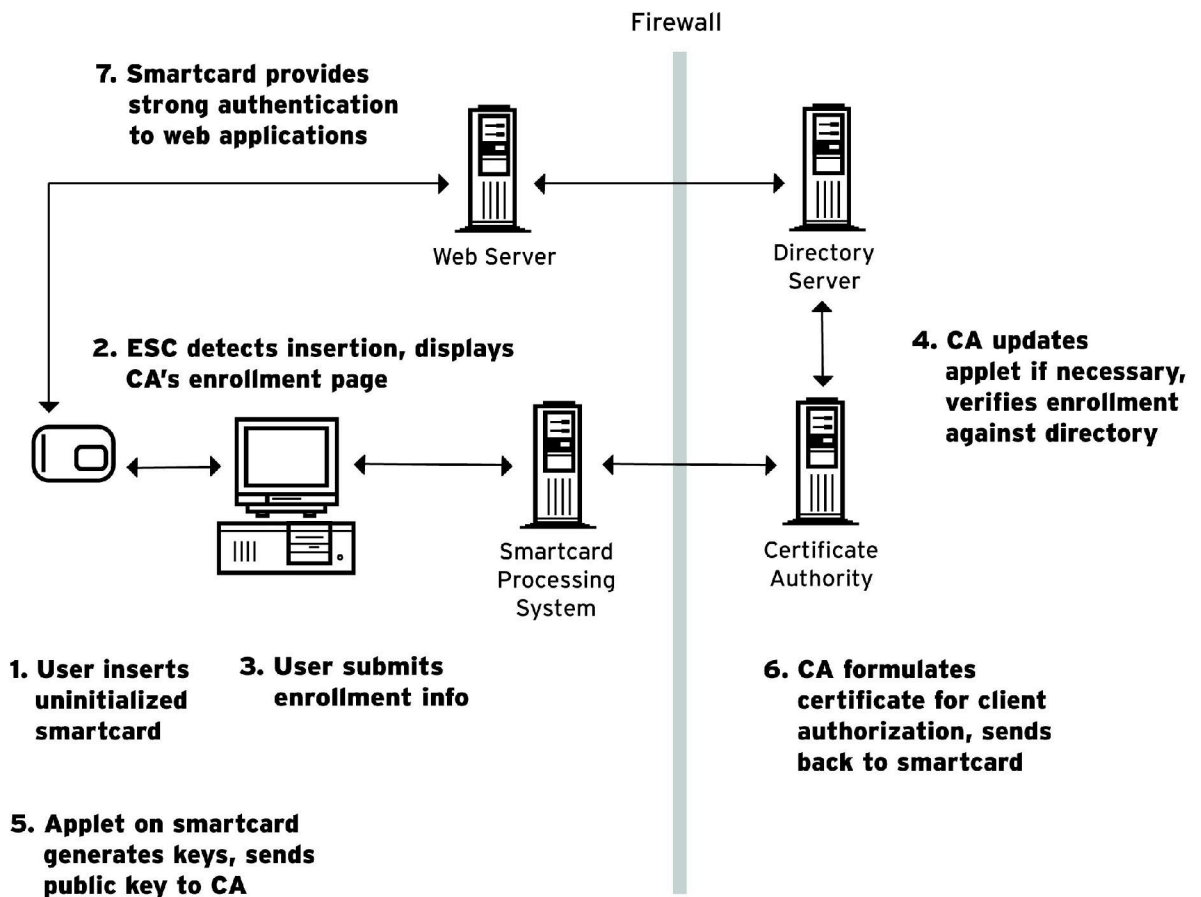
After publishing the certificate to the directory for verification purposes when the user authenticates to a website, the CA sends the signed certificate back to the user's computer for storage with the private key in the browser's certificate database.

This relatively low-cost scenario is designed for issuing "soft" certificates for use on a single machine. Deployment costs are lower than for more complex deployments involving smartcards or private key archival. Red Hat Certificate System includes tools for generating one-time PINs if needed.

The level of assurance is better than repeated use of a name and password, since client authentication does not require sending a secret over the wire. This level of assurance may be appropriate for some customers or intranet users, but is lower than the assurance provided by smartcards.

**Enrollment scenario: Partner identities, medium assurance**

This scenario shows a partner or extranet user obtaining new PKI credentials for a smartcard. As with the first scenario, the entire interaction, including both enrollment related administration, occurs remotely. The main difference is the enhanced security and portability provided by smartcards.



Depending on the organization's processes, the entire enrollment procedure can take less than a minute, including the time it takes the user to type the requested information:

1. The user inserts an uninitialized smartcard distributed by the organization running Red Hat Certificate System.

2. The Enterprise Security Client on the user's machine detects the insertion event, contacts the CA, and fetches a custom HTML window designed by the organization to meet its initial enrollment requirements.
3. The user fills in the requested information, assigns a PIN to the smartcard, and clicks the *Submit* button.
4. The CA checks the version of the applet and updates it if necessary, then verifies the enrollment information against the Red Hat Directory Server or any other systems required by the organization's policies.
5. The applet on the smartcard generates the public and private keys and submits the public key to the CA. The private key remains on the card.
6. The CA formulates a certificate, publishes it to the directory, and sends it to the smartcard.
7. The user is now ready to use strong SSL client authentication

The entire process takes only a minute or two. The partner is now ready to use the smartcard for strong SSL client authentication to extranet web sites.

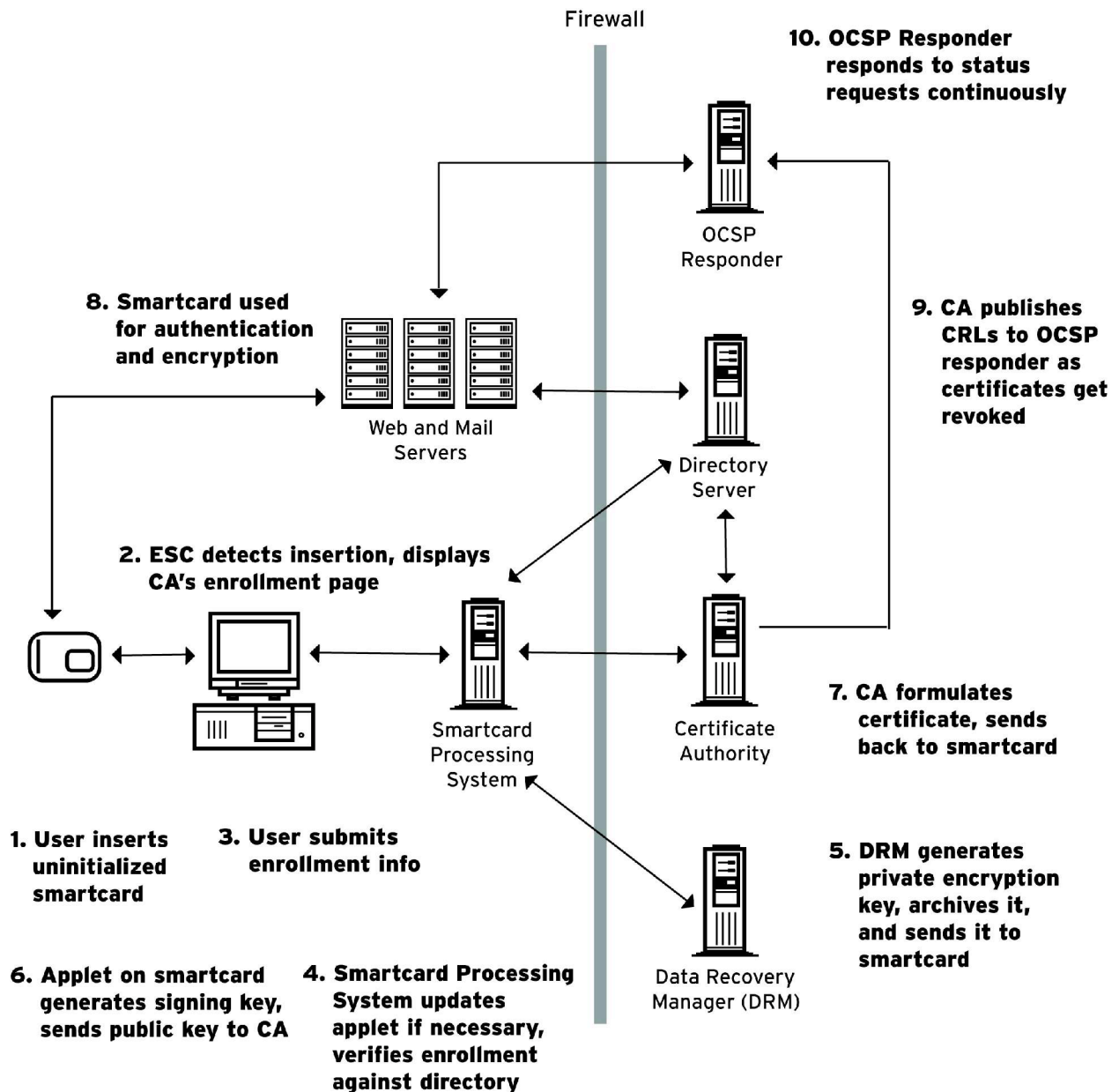
The smartcard is not bound to a specific person until someone uses it to go through the enrollment process. The organization issuing the credentials can customize the enrollment information according to whatever level of assurance its policies require. For example, a phone call or one-time password might be involved.

Stealing the uninitialized card doesn't help a potential impostor unless the enrollment depends only on information that can be found with the card. The organization planning the deployment must evaluate the security threat and require appropriate information and verification to complete the enrollment. Secure communication between the Certificate Authority and the smartcard is achieved through the use of symmetric keys and the Protocol Data Units (PDUs) defined by the Global Platform smartcard protocol.

This enrollment scenario is designed to provide strong authentication to websites at a relatively low deployment cost. If an organization uses PKI credentials for digitally signed and encrypted email and other high-security encryption tasks that require private key archival, the next scenario provides the highest level of assurance and management flexibility.

**Enrollment scenario: Employee identities, high assurance**

This scenario shows an employee obtaining new high-security PKI credentials for a smartcard.



This scenario could potentially be integrated with a more complex enrollment/registration process involving face-to-face interaction, collection of biometric information, and background checks as well as issuance of smartcard credentials.

The process illustrated on the previous page works like this:

1. The user inserts an uninitialized smartcard distributed by the organization running Red Hat Certificate System.
2. The Enterprise Security Client on the user's machine detects the insertion event, contacts the CA, and fetches a custom HTML window designed by the organization to meet its initial enrollment requirements.
3. The user fills in the requested information, assigns a PIN to the smartcard, and clicks the Submit button.
4. The Smartcard Processing system acts as a traffic cop for communication between the back end and the smartcard. At this stage, it checks the version of the applet and updates it if necessary, then verifies the enrollment information against the Directory or any other systems required by the organization's policies.
5. The key archival system (DRM in the figure) generates the encryption key pair, archives the private encryption key, and sends it securely to the applet on the smartcard along with the public key.
6. The applet on the smartcard generates the signing key pair and sends public key to the CA in a certificate request.
7. The CA formulates a certificate, publishes it to the directory, and sends it to the smartcard.

The user is now ready to use strong SSL client authentication.

Steps 8, 9, and 10 illustrate the ongoing process of certificate usage and status checking. Each time a user signs or encrypts an email or authenticates to a server, the application involved checks the status of the certificate with the OCSP Responder.

The CRL used by the OCSP Responder is continuously updated by the Certificate Manager as certificates get revoked.

As in the previous scenario, secure communication between the Certificate Manager and the smartcard is achieved through the use of symmetric keys and the Protocol Data Units (PDUs) defined by the Global Platform smartcard protocol.

This enrollment scenario is designed to provide strong authentication and encryption for high-security applications requiring a high level of assurance and a reliable system of private key archival. It could potentially include multiple OCSP responders, cloned globally distributed CAs and sub-CAs, as well as third-party hardware security modules (HSMs) and other specialized hardware.

## **Technical highlights: Red Hat Certificate System**

### **Architecture**

These are the main components of the Red Hat Certificate System:

- **Certificate Authority (CA):**
  - Issues X.509 digital certificates and CRLs
  - Publishes certificates to the LDAP directory.
- **Token (Smartcard) Processing System (TPS):**
  - Supports Global Platform smartcards & software tokens
  - Functions as the registration authority with which the client communicates directly
- **Data Recovery Manager (DRM):**
  - Secure repository for backup/recovery of user's private keys
  - Configurable multi-person approval for recovery
- **Online Certificate Status Protocol (OCSP) Responder:**  
Responds to OCSP requests to verify certificate validity in real time

- **Token Key Service (TKS):** Manages symmetric keys for securing communication between subsystems and smartcards
- **Enterprise Security Client (ESC):** Client software for Red Hat Enterprise Linux, Windows XP, and Mac OS X.

See the Enrollment Scenarios above for more information about interactions among these components.

### **Smartcard Innovations**

In recent years, Red Hat Certificate System development has focused on simplifying enrollment and other aspects of smartcard management. It is the first product that provides a completely integrated end-to-end PKI and smartcard management solution.

During enrollment, the Enterprise Security Client displays an HTML page controlled by the back end. All aspects of enrollment are completely customizable. This flexibility allows organizations to adjust the level of assurance (and budget) required for enrollment according to the changing security requirements of each resource or role.

Once the user has activated enrolled the smartcard, he or she can immediately start using it to authenticate to websites or sign and encrypt email. Red Hat engineers have contributed code to open source applications such as Firefox and Thunderbird so that they can detect smartcard insertion and removal events and respond appropriately.

Red Hat is also planning to integrate ESC and related smartcard management capabilities with future versions of Red Hat Enterprise Linux®.

### **Online certificate status checking**

High-assurance deployments of Red Hat Certificate System need to provide a way of checking that a given certificate is still valid each time it's used.

One way PKI products support this requirement is through Certificate Revocation Lists (CRL)s, which are lists of certificates

issued by a CA that have been revoked or are no longer valid for some other reason. Red Hat Certificate System supports CRLs. However, CRLs can become quite large, which tends to slow down the responsiveness of applications attempting to use them.

The Online Certificate Status Checking (OCSP) protocol allows OCSP-compliant applications to determine the state of a certificate, including the revocation status, without having to directly check a CRL. Instead, the CRL is published by a CA to a validation authority, which is also called an *OCSP responder*. The OCSP responder typically does the checking for the application more quickly than directly checking a CRL.

The OCSP responder provided by Red Hat Certificate System provides a high response rate sufficient for most deployments.

### **Scalability and performance**

Red Hat Certificate System incorporates Red Hat Directory Server as its certificate store, so it can take full advantage of that product's superb scalability and performance.

In addition to supporting very large high-assurance deployments in government and industry for many years, Red Hat Certificate System has demonstrated the following performance in lab tests

- Issued over 12 million certificates from single server in less than 35 days (~14,000 certificates/hour)
- Simultaneously published to Directory Server and archived private keys
- Revoked 10% of certificates, resulting in 1.2-million-entry CRL
- Generated CRL in less than 30 minutes

### **High availability and disaster recovery**

Red Hat Certificate System uses the Red Hat Directory Server internally to facilitate a robust cloning and failover architecture. The CA, DRM, and OCSP Responder can all be cloned. This feature almost completely eliminates unplanned outages by making one or more subsystem clones available for failover.

Data sources for cloned systems are replicated, so data is shared seamlessly between subsystem databases. Master and cloned instances typically installed on different machines behind a load balancer. When a failure occurs, the load balancer transparently redirects all requests to a clone that's still running, without any service interruption.

### **Tools and SDKs**

Red Hat Certificate System includes a Java SDK to facilitate integration of a PKI infrastructure with other enterprise applications. This includes extensive documentation for creating plug-ins, for example to bootstrap existing authentication mechanisms or trigger billing when a certificate is published.

To perform routine management tasks remotely, administrators can use either the Red Hat Management Console or an extensive array of command-line administrative and testing tools.

### **Government support**

An earlier version of the Network Security Services (NSS) crypto engine, used in both Red Hat Directory Server and Red Hat Certificate System, was certified under FIPS 140-1 levels 1 and 2. FIPS 140-1 Level 3 requires an external HSM. Red Hat is engaged in ongoing HSM certification with several vendors. In mid-2005, FIPS 140-2 recertification for NSS is well underway.

An earlier version of Red Hat Certificate System was NIAP Certified under the Common Criteria CIMC (Certificate Issuance and Management Components) Protection Profile, evaluation level 4 Augmented. In mid-2005, preparation for Common Criteria recertification is beginning.

Red Hat Certificate System has been used to issue certificates accepted by Federal Bridge, the gateway mechanism used by government agencies to ensure that they can recognize each other's CAs. The current product fully supports the Federal Bridge technical requirements, including the policy extension and, for the CA certificate, the policy mappings extension.

Red Hat Certificate System complies with the standards published by the Public Key Infrastructure (X.509) (PKIX)

working group and supports certificate issuance with Windows extensions for Windows Smartcard Logon, both of which are required for many government deployments. To comply with the GSA E-Authentication Initiative, Red Hat is also planning to add SAML support to future releases.

For more information, visit [redhat.com](http://redhat.com) or contact us at 1888REDHAT1( US and Canada) / +19197543700 (international).