

SecureAuth Authentication: *How SecureAuth performs what was previously impossible using X.509 certificates*

As enterprises move their applications to the Web and mobile platforms, providing strong security for those applications becomes more critical. An enterprise must gauge an array of factors prior to implementing a solution that it deems viable and secure. Chief among these factors are strong security, ease-of-use for end-users, deployable technology, maintenance and cost.

Traditionally, Public Key Infrastructure (PKI) has been regarded as the strongest form of authentication. However, PKI's promise has never stretched beyond government agencies and organizations that can boast the infinite budgets and resources needed to manage PKI. PKI technology has been stagnant because of the fact that it is; complex to deploy, cost-prohibitive, difficult to manage and use. Also, key elements of its technology (specifically usability and interoperability) have not evolved along with modern applications and the Internet. For these reasons, PKI has failed to be widely adopted by most enterprises.

SecureAuth distinguishes itself from PKI by delivering X.509 v3 technology that is not only more secure than traditional PKI, but infinitely more feasible. **SecureAuth** is a solution founded on the principle of providing strong, 2-factor authentication that is affordable, easy for end-users to use, and easy for administrators to deploy and maintain.

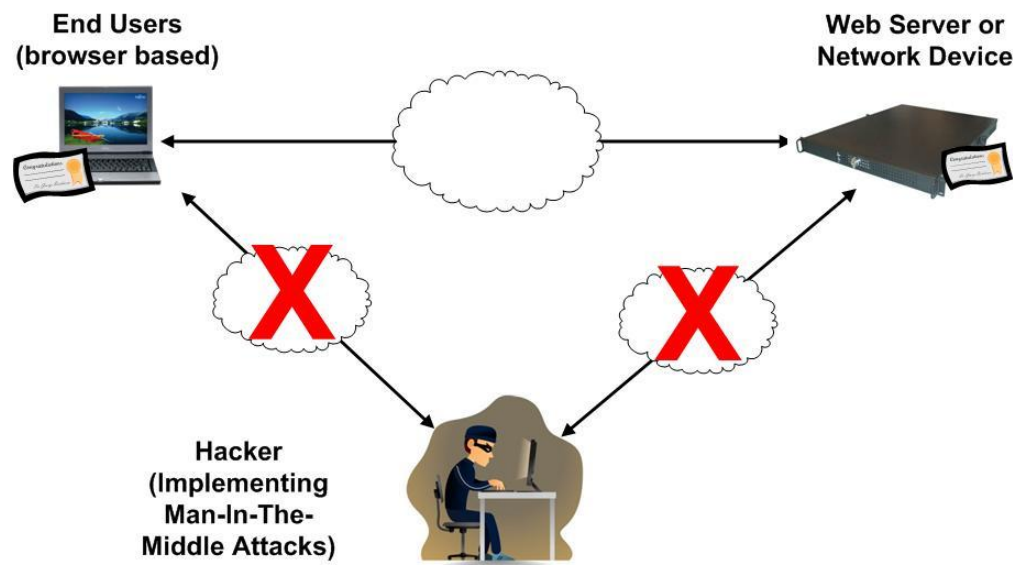


Figure 1: X.509 v3 Technology is proven to prevent man-in-the-middle identity attacks.

SecureAuth® is Easy to Deploy

Traditional PKI requires several infrastructure components to function:

- Redundant set of certificate servers
- Redundant set of certificate revocation lists
- Data store of certificates issued
- Set of servers for out-of-band registration

In addition, if the “enforcement point” is a web server, then the web server must be converted to allow Client Side SSL (C-SSL). This is not a trivial task and breaks many pre-existing applications (See Figure 2).

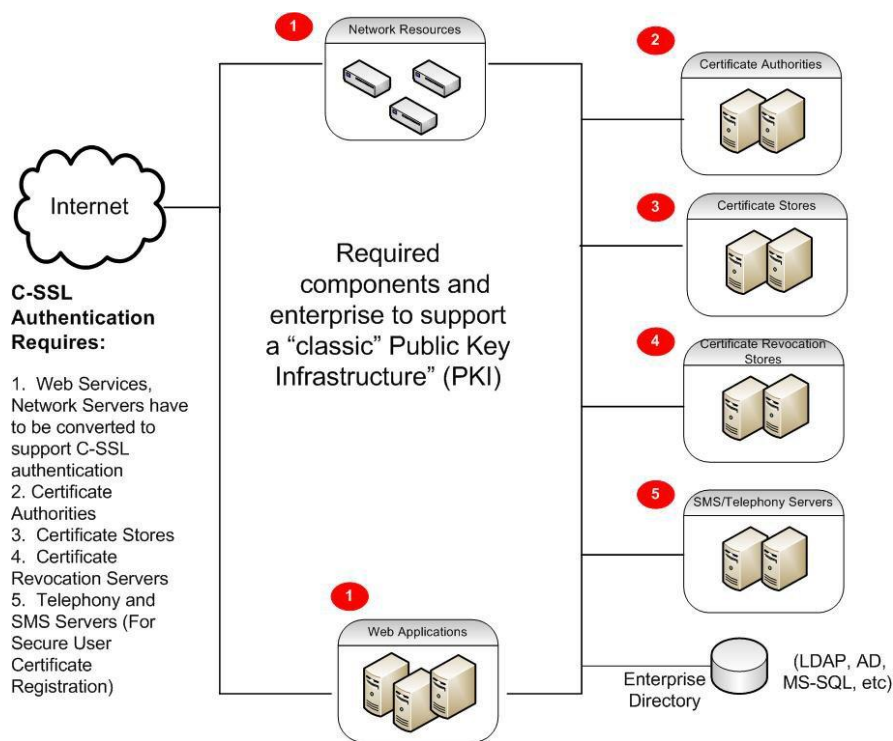


Figure 2: PKI has an onerous amount of infrastructure for an enterprise to install and maintain.

SecureAuth acknowledges that this infrastructure is burdensome on an enterprise and **SecureAuth** does NOT require these components for an installation.

SecureAuth installs an authentication appliance that meets the requirements of the above components without the overhead. **SecureAuth** is able to meet these functional requirements through two key architectural differences; native data store and secure web services.

SecureAuth Utilizes the Native Data Store

The **SecureAuth** authentication appliance is shipped with data store connectors that enable it to integrate into an enterprise’s native data store. This ability to use the “data store of record” means that there is no data syncing, and no need for certificate retention and certificate revocation list (See Figure 3).

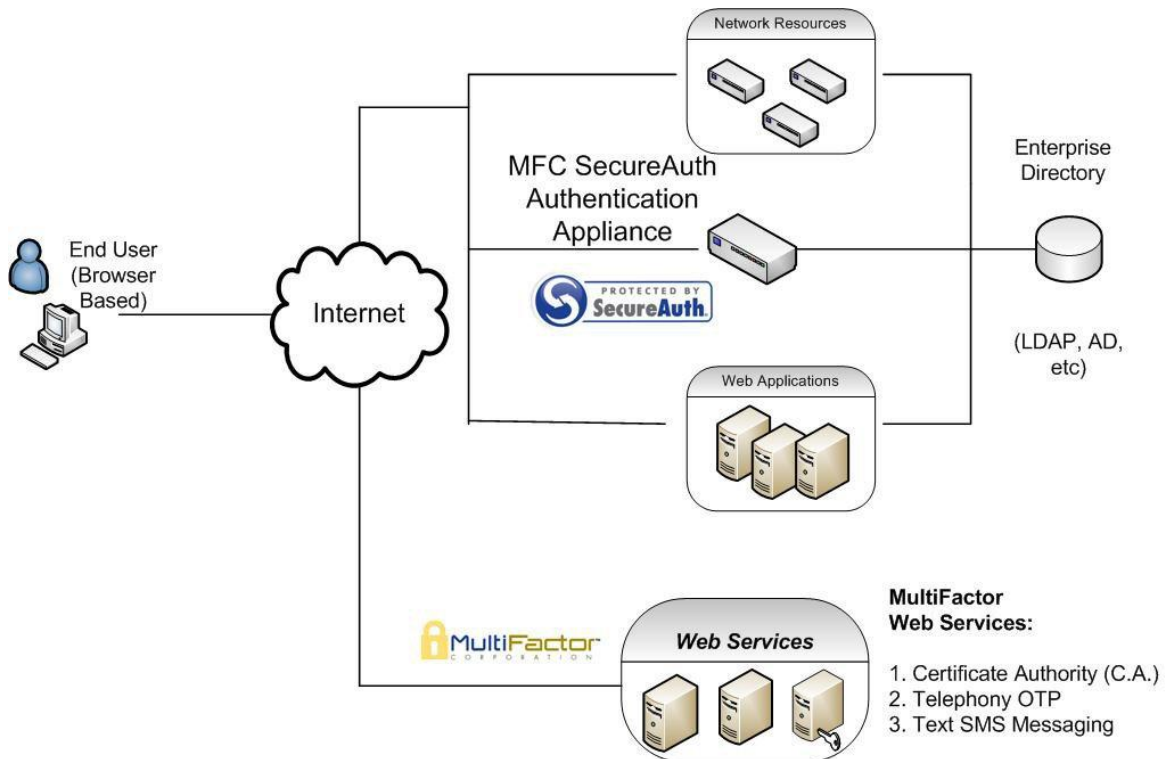


Figure 3: **SecureAuth** in a deployed environment for network and web authentication.

SecureAuth Utilizes Secure Web Services

Upon deployment, the **SecureAuth** authentication appliance is enabled to utilize **SecureAuth’s** hosted web services for key functional needs. The **SecureAuth** web services include:

- Telephony One-Time-Registration Passwords
- SMS One-Time-Registration Passwords
- Certificate Servers

These web services allow for a simple installation and integration of the **SecureAuth** authentication appliance into a network and web deployment (See Figure 3).

The secure, WSE 3.0 authenticated web services insure that an enterprise does not have to deploy additional infrastructure and thus saves the cost and expense of the initial installation and maintenance of these components.

SecureAuth is Easy to Maintain

PKI is the only technology in the world that requires:

- A list of all valid credentials
- A list of all invalid credentials (“anti-database”)

It is this second list, the “anti-database” that has made PKI all but impossible to maintain.

This “anti-database” is actually built into the fundamental of certificates. A certificate has subject data such as a username, expiration date, public/private key-pair and signature from a trusted Certificate Authority. The validity of a certificate is based on the expiration date and the signature. A certificate that has not expired and is properly signed by an authority you trust is considered a valid username/password combination.

You can see how this system can rapidly break down. Once a certificate is issued, it will be granted access until it expires. What happens when the person associated with the certificate leaves the organization? It is not always feasible to retrieve the certificate, so there must be some other mechanism to exclude previously issued certificates. Hence PKI came up with the “anti-database” technology to compensate for this.

The dates of these “anti-database” technologies are illuminating:

- CRL – Certificate Revocation List ([RFC 1422](#)) 1993
- OCSP – Online Certificate Status Protocol ([RFC 2560](#)) 1999

Both of these technologies were devised and implemented before enterprise directories were implemented and available as the data store of record. PKI, when previously implemented, had little or no relation to the central location where an enterprise manages user-accounts. To lock an account, the additional step of revoking the certificate was a requirement.

Systems that rely on certificates for identity information and to validate access to systems or networks, create mechanisms to “authorize” access against the data store of record. But in the end, a valid certificate is still equated to a passed authentication.

Even with authorization against a central store, enterprises often required these CRL and OCSP architectures, which have proven to be unwieldy and un-deployable to anything but limited B2E network deployments. There has been no market acceptance of CRLs and OCSP technologies for B2C and web environments because of the cost and complexity associated with the solution.

SecureAuth leverages the advances in technology since 1999, namely the central data store that now exist in virtual every enterprise today (See Figure 3). **SecureAuth** enables an enterprise to set a “Per User Certificate Limit” in the **SecureAuth** admin console. This is configurable per **SecureAuth** instance (See Figure 4).

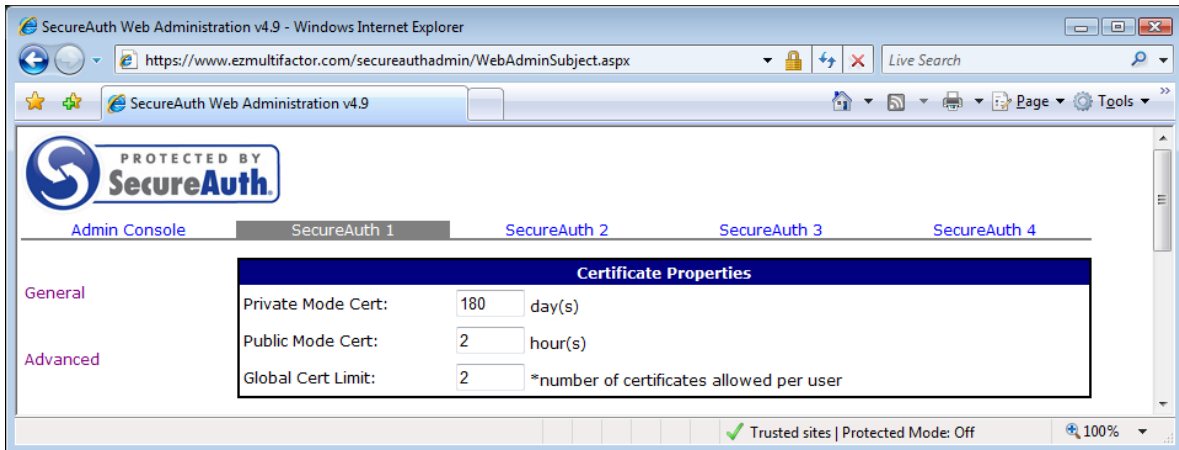


Figure 4: **SecureAuth** allows an enterprise to configure number of certificates per user.

SecureAuth, because it is an abstracted authentication appliance (See Figure 3) is able to inspect two sets of information upon every new authentication and credential information. These two sets of information are:

- Instance certificate limit for the resource (See Figure 4)
- Current Certificate Issued, per user (See Figure 5)

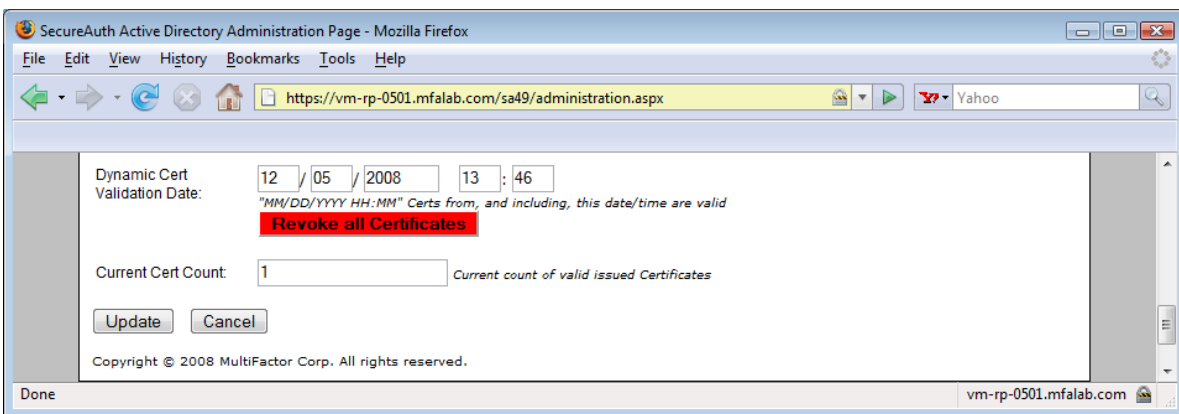


Figure 5: **SecureAuth** keeps track of the certificates issued to the user and allows “one-touch” revocation with the Cert Revocation Button.

Because **SecureAuth** is the authentication enforcement point, **SecureAuth** is able to inspect current issued certificates and choose NOT to distribute subsequent certificates based on this count.

Just as important, a **SecureAuth** administrator is able to revoke the validity of certificates with the action of a single button (**Revoke all Certificates**) on the **SecureAuth** console (See Figure 5). Invoking this feature sets:

- Validation Date to the current time (See Figure 6)
- Certificate Count to 0 (See Figure 6)

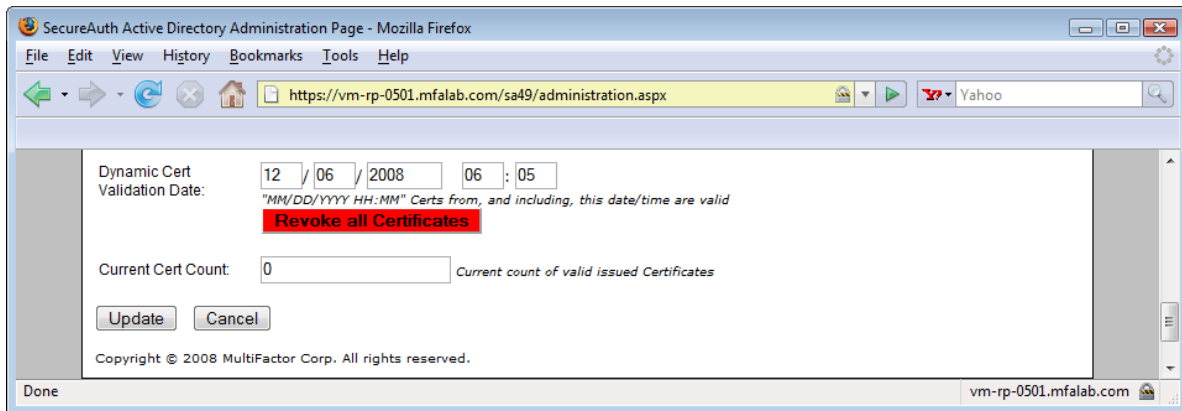


Figure 6: A single button, “The Revoke All Certificates,” revokes the certificates for the user without CRLs or OCSPs.

The key here, and unique to the **SecureAuth** solution, is that **SecureAuth** Authentication appliance is the authentication enforcement point (See Figure 3). **SecureAuth** does not have to rely on the “kludgy” and hard-to-manage CRLs and OCSP infrastructure to enforce the revocation – **SecureAuth** is the enforcement point and thus can inspect and choose not to accept the credential. This decision can be based on any attribute of the certificate itself, data from the data store or a combination of both.

SecureAuth is Easy to Scale

Most enterprises are very comfortable with deploying high availability IT resources. The concept of load redundant web servers and load balancers is commonplace in the modern IT infrastructure.

Unfortunately, Certificate Authorities (C.A.s), the core component of the PKI infrastructure, does not lend itself to this type of scaling. Unlike web servers, the concept of simply adding another server and the placing a load balancer in front of the set of certificate servers simply will not work. In addition, Certificate Authorities do not have any concept of clustering.

Creating a high availability C.A. infrastructure is not an activity that most IT enterprises have the talent or resources to create. The knowledge required to create a Trusted Root Certificate Authority and then create a set of intermediate certificate authorities signed with the Trusted Root Certificate, is not common place in today's environment. The cost associated with this skill set is prohibitive for a high availability deployment of PKI to most enterprises.

SecureAuth alleviates this cost set by removing this costly infrastructure from the enterprise (See Figure 8). A secure WSE 3.0 Web Service call is made between the **SecureAuth** Authentication and the **SecureAuth** Web Service (See Figure 8). Because this call is X.509 v3 authenticated web service connection, and requires the enterprise to have a registered **SecureAuth** server to access the web service.

Creating a set of certificate authorities residing behind a protected set of web services is not a trivial task and is beyond the scope of most enterprises. **SecureAuth** has created a unique set of high-availability certificate authorities behind a protected set of web services (See Figure 8). This is an industry unique configuration that can:

- Securely create/distribute X.509 v3 credentials without infrastructure
- Scale their authentication to infinite user counts

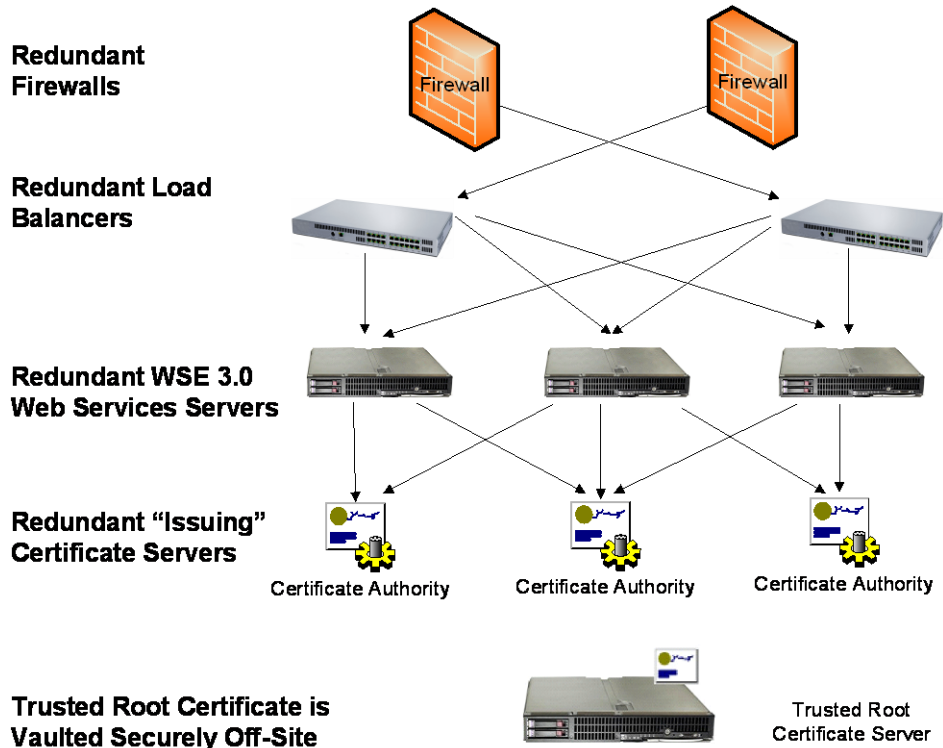


Figure 7: **SecureAuth** hosts a high availability infrastructure of certificate authorities authenticated by WSE 3.0 web services. For large deployments (100,000 plus users) **SecureAuth** can assist enterprises to create this infrastructure.

It is important to note that different end user X.509 v3 certificates created from this infrastructure contain information ONLY germane to the specific enterprise deploying a **SecureAuth** authentication appliance. A certificate created for enterprise A is not necessarily valid for enterprise B, despite being issued from the **SecureAuth** infrastructure. **SecureAuth** is able to do this via an advanced set of policy servers executing policy on the **SecureAuth** Certificate Authorities. The information is "tagged" in the PKCS #10 requests sent from the **SecureAuth** appliance to the **SecureAuth** Web Service C.A. (See Figure 3). The result is an end user certificate with OU and DC information in the end user certificate that maps only to the deploying **SecureAuth** authentication site.

SecureAuth Allows for Easy Application Integration

For PKI deployments, the act of creating a PKI infrastructure and then deploying the certificates does not complete the task for the enterprise – the enterprise now has to determine how to make its current application and network resources accept an X.509 v3 authentication.

PKI for applications requires web services to “turn-on” Client side SSL authentication (C-SSL). This is not a trivial activity (Microsoft Outlook Web Access has a 32-page documentation on the process). It is NOT a matter of just re-writing the authentication page. The actual web server or virtual web server has to be configured to conduct a C-SSL authentication. This is a global setting that has ramifications across all applications that are hosted on the web server. The process of turning on and supporting C-SSL on web servers is NOT the same across disparate web servers (such as Microsoft IIS, Apache, IBM HTTP-D and Zeus).

SecureAuth Solves the Application Integration Issue

Applications do not have to convert the web server to enable C-SSL authentication to utilize the strength of **SecureAuth’s** bilateral X.509 v3 authentication. Unlike C-SSL, **SecureAuth** is able to utilize the native target/redirect and sessioning mechanisms of the application (*See Figure 8*).

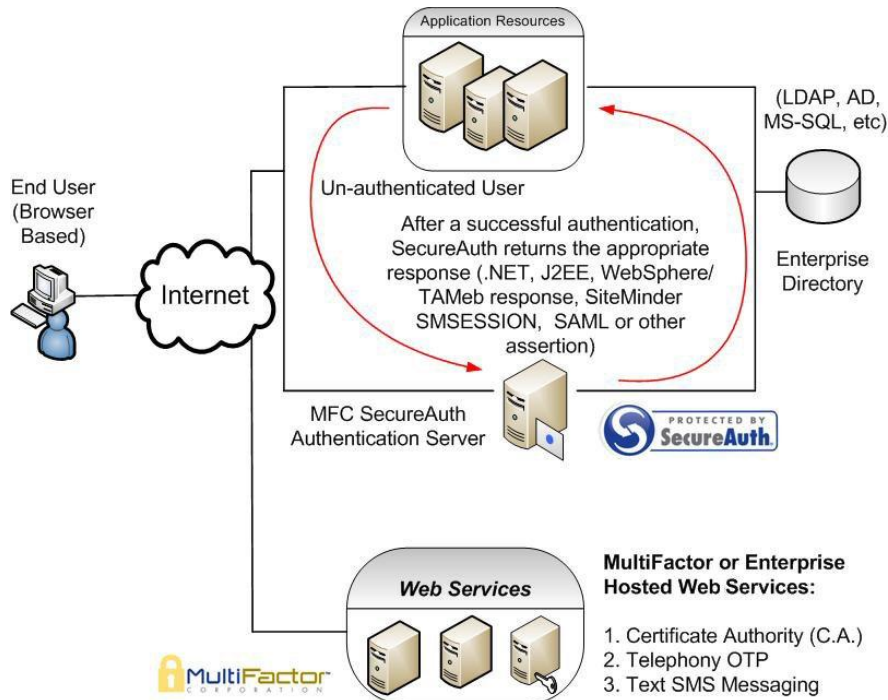


Figure 8: The **SecureAuth** has the unique ability of integrated directly with the application requiring X.509 v3 authentication without forcing the enterprise to “turn-on” C-SSL on the web server.

SecureAuth integration to an existing application can be as simple as turning on Forms Based Authentication and directing the resource to the **SecureAuth** server. This is the case for applications such as Microsoft SharePoint/MOSS, Outlook Web Access and other ASP.NET applications. Similar target/redirect methodologies exist for J2EE applications and application servers such as IBM’s WebSphere.

SecureAuth is currently integrated in the following environments:

- ASP.NET applications
- Microsoft Outlook Web Access
- Microsoft SharePoint/Moss applications
- J2EE applications
- CA SiteMinder
- Tivoli Access Manager for e-Business (TAMeb)
- Applications that can take a SAML (1.1 and 2.0) assertion
- Applications that accept OpenID authentication
- Any application that can read a GUID in share datastore
 - **SecureAuth** has utilized this mechanism for applications in the “other” category

In addition, **SecureAuth** uniquely facilitates an incremental approach to federation. **SecureAuth** becomes the authentication mechanism for the Identity Provider (IdP). Once implemented, **SecureAuth** or another mechanism can create the SAML , OpenID, WS-* or other assertion that the Relying party is expecting (See Figure 9).

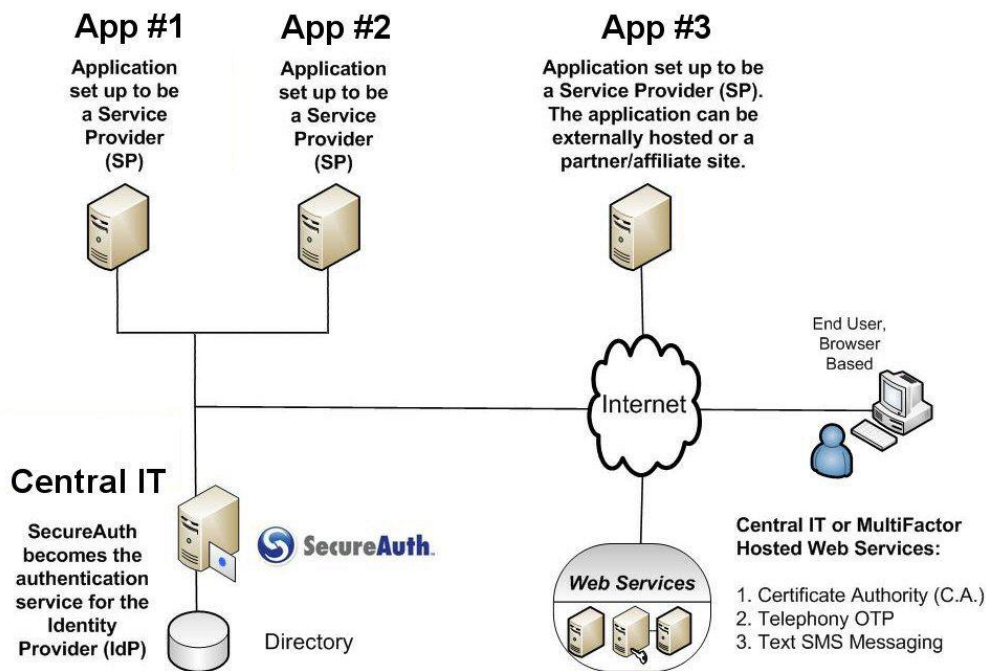


Figure 9: **SecureAuth** can be the Authentication Mechanism for the Identity Provider in a federated environment.

SecureAuth Allows End Users to Easily Obtain Credentials

One of biggest “deal killers” of a PKI deployment is the difficulty associated in delivering the X.509 v3 credentials to the end user. One central issue has been in educating end users on what mechanism for:

- Requesting a public/private key
- Storing the public/private key
- Transporting the public/private key

Traditional PKI does not have an answer for these problems. End users simply do not have PKI backgrounds or the understanding of the nuances and complexities of private and public keys. Attempts to put the credentials on portable key store frequently fail because of the lack of knowledge on the export and import process.

SecureAuth addresses all of these issues with:

- User Self-Service Registration (*See Figures 10-14*)
- One-Touch User Certificate Revocation (*See Figures 5 and 6*)

The **SecureAuth** user self-service registration process is self-explanatory and requires no helpdesk support. Users are required to complete this once to acquire an X.509 v3 credential. *Figures 10-14* walk through the process:


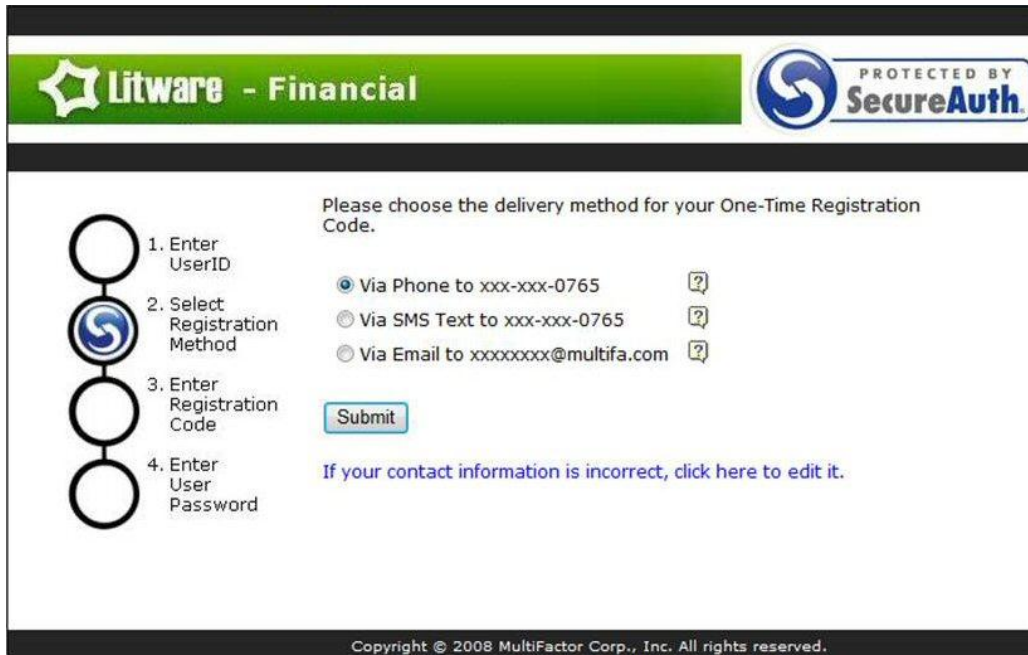




Figure 10: User enters their User ID to begin the registration process.





Litware - Financial  PROTECTED BY **SecureAuth**

Please choose the delivery method for your One-Time Registration Code.

1. Enter UserID
2. Select Registration Method
3. Enter Registration Code
4. Enter User Password

Via Phone to xxx-xxx-0765 

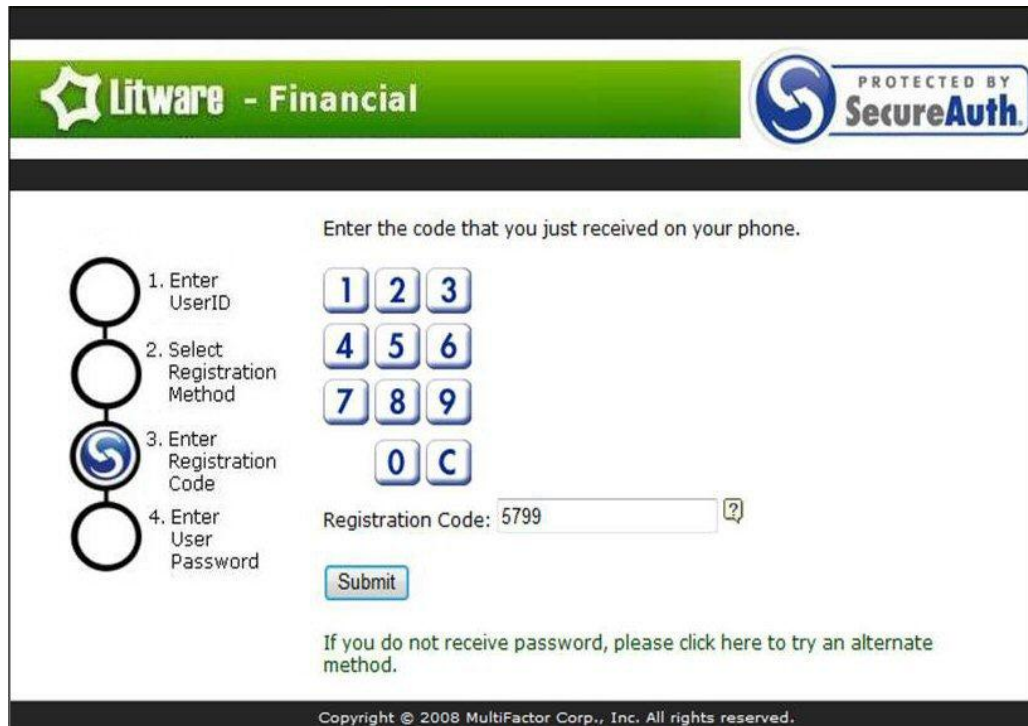
Via SMS Text to xxx-xxx-0765 


Via Email to xxxxxxxx@multifa.com 

[If your contact information is incorrect, click here to edit it.](#)

Copyright © 2008 MultiFactor Corp., Inc. All rights reserved.

Figure 11: The user selects a registration method from the (enterprise-configurable) list of options.



Litware - Financial  PROTECTED BY **SecureAuth**

Enter the code that you just received on your phone.


1. Enter UserID
2. Select Registration Method
3. Enter Registration Code
4. Enter User Password

1 2 3

4 5 6

7 8 9

0 C

Registration Code: 5799 

[If you do not receive password, please click here to try an alternate method.](#)

Copyright © 2008 MultiFactor Corp., Inc. All rights reserved.

Figure 12: The user enters the One-Time-Registration Code via Java Keypad.

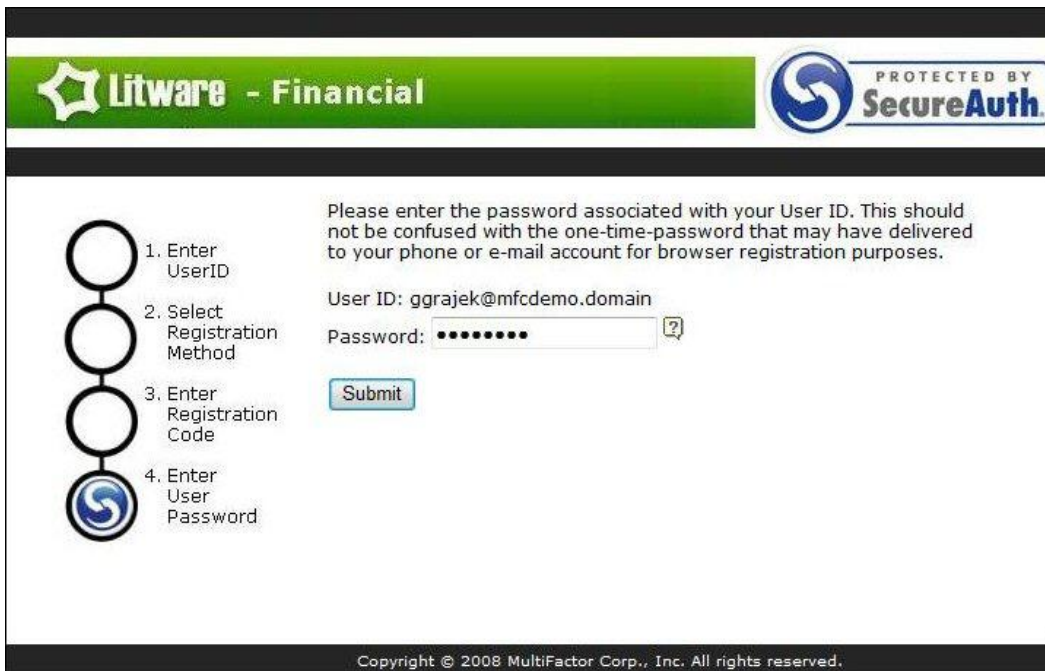


Figure 13: User inputs his enterprise (AD, LDAP MS-SQL, etc) password.

Note: The password is stored in the enterprise data store and not replicated.

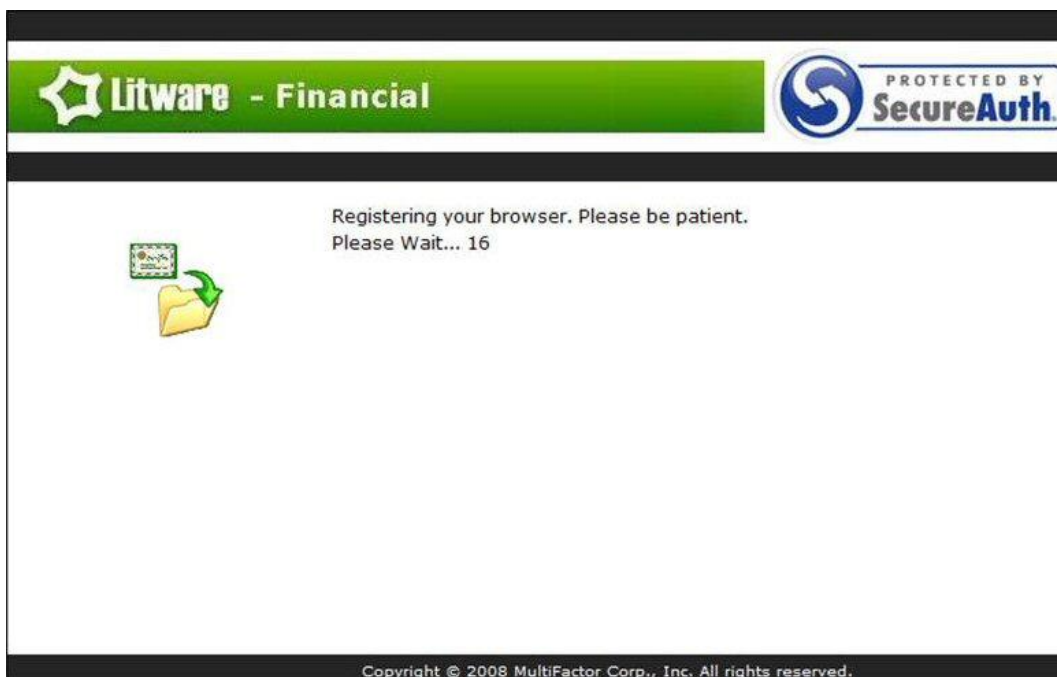
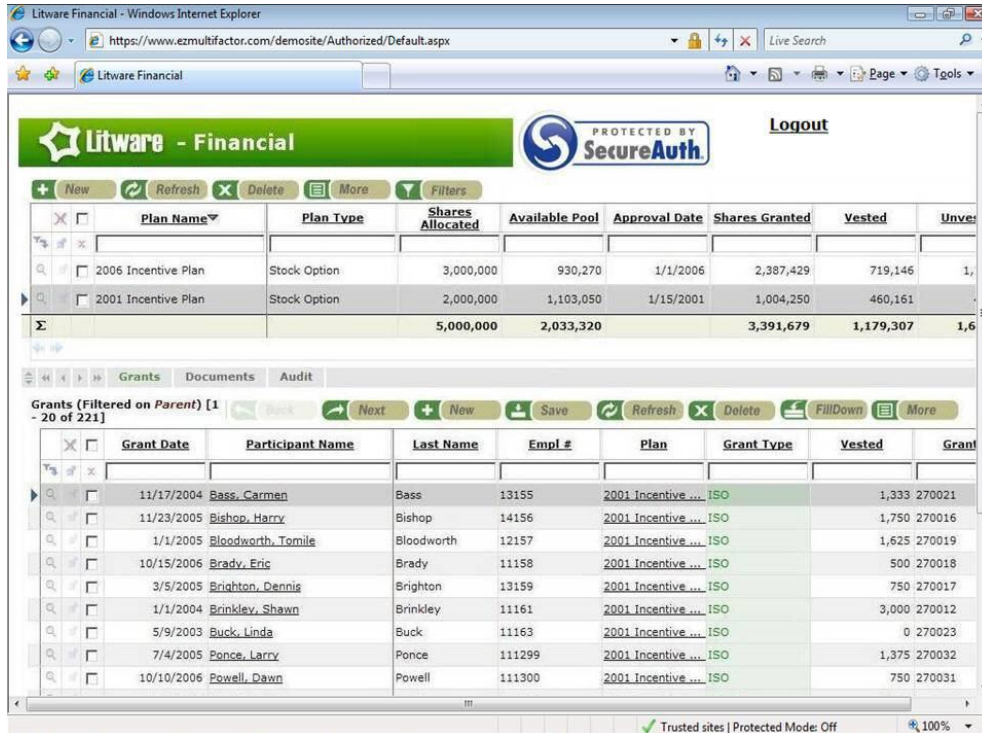


Figure 14: *SecureAuth* creates the public/private key pair on the end users device. (The *SecureAuth* private key never crosses the network.)



Plan Name	Plan Type	Shares Allocated	Available Pool	Approval Date	Shares Granted	Vested	Unvested
2006 Incentive Plan	Stock Option	3,000,000	930,270	1/1/2006	2,387,429	719,146	1,668,283
2001 Incentive Plan	Stock Option	2,000,000	1,103,050	1/15/2001	1,004,250	460,161	544,089
Σ		5,000,000	2,033,320		3,391,679	1,179,307	1,668,283

Grant Date	Participant Name	Last Name	Empl #	Plan	Grant Type	Vested	Grant
11/17/2004	Bass, Carmen	Bass	13155	2001 Incentive ...	ISO	1,333	270021
11/23/2005	Bishop, Harry	Bishop	14156	2001 Incentive ...	ISO	1,750	270016
1/1/2005	Bloodworth, Tomile	Bloodworth	12157	2001 Incentive ...	ISO	1,625	270019
10/15/2006	Brady, Eric	Brady	11158	2001 Incentive ...	ISO	500	270018
3/5/2005	Brighton, Dennis	Brighton	13159	2001 Incentive ...	ISO	750	270017
1/1/2004	Brinkley, Shawn	Brinkley	11161	2001 Incentive ...	ISO	3,000	270012
5/9/2003	Buck, Linda	Buck	11163	2001 Incentive ...	ISO	0	270023
7/4/2005	Ponce, Larry	Ponce	111299	2001 Incentive ...	ISO	1,375	270032
10/10/2006	Powell, Dawn	Powell	111300	2001 Incentive ...	ISO	750	270031

Figure 15: In the final step, the user is redirected to the Application

SecureAuth handles the creation of the public/private key for the end user. The self-explanatory, user self-service registration process takes all the guesswork out of the registration process.

SecureAuth is True 2-Factor

One of the primary security concerns that eliminate PKI from authentication decisions is that PKI is actually only single factor authentication. The argument is that the choices for “factors” are:

- Something you have
- Something you know

PKI, the authentication factor is only the first, something you have. And the security argument is: “if the device is compromised or stolen” there is no other factor since the certificate is the only factor.

There are new “PKI” solutions on the market being offered. These solutions promise to add a factor into the equation by utilizing a password in the “unlocking” of the key store. Security analysts have accurately stated that these solutions are not true 2-factor authentication. The fact that the password is locked with the credential simply means the solution is purely 1-factor and the attack method is simply a “brute force” attack on the credential to “guess” the password. In this manner, the security analyst are correct – these solutions are simply 1-factor since the certificate can be removed from the device and then brute force attacked. Once the certificate is unlocked from these means, the credential can be replayed and the identity of the legitimate user impersonated by the hacker.

SecureAuth is different from both PKI and these “PKI solutions with a twist.”

SecureAuth is unique in the X.509 v3 authentication market by its ability to conduct its own validation of the X.509 v3 credential. That is, **SecureAuth** is able to conduct the public/private key pair exchange that occurs in the **SecureAuth** authentication dialogue. **SecureAuth** does not rely on legacy PKI code for validation – instead it has the application redirect the authentication to the **SecureAuth** authentication appliance for the authentication dialogue (*See Figure 16*).

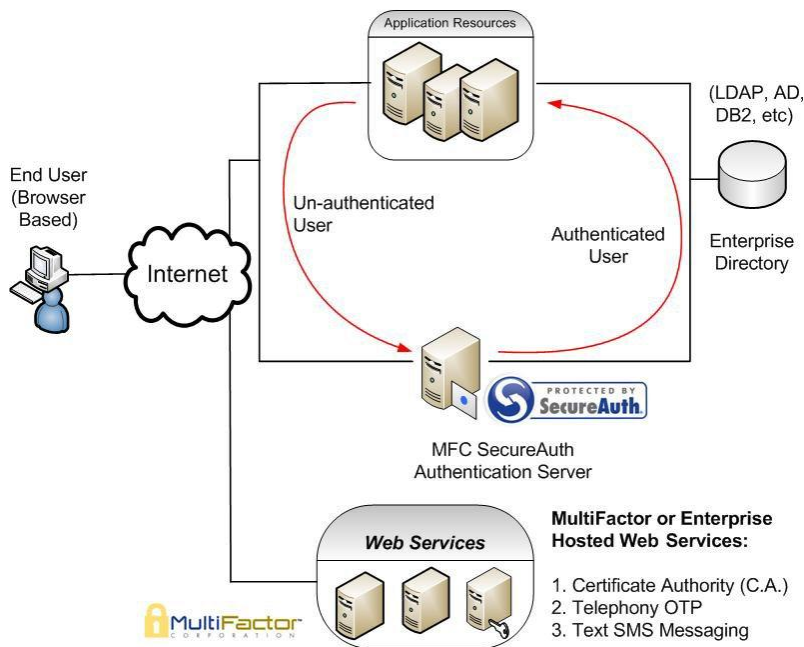


Figure 16: **SecureAuth** conducts its own client/server validation.

Because **SecureAuth** is not relying on legacy, circa 1988 validation code, **SecureAuth** is able to conduct a true 2-Factor, 2-way authentication, utilizing the original and valid public and private key infrastructure. Since, the **SecureAuth** authentication is conducting the authentication – a second factor authentication is enforced.

The **SecureAuth** First Factor X.509 v3 Authentication

In a **SecureAuth** X.509 v3 validation, the user's X.509 v3 credential (the first) factor is validated via the following means:

- User's public Key Sent to **SecureAuth** Server
- Certificate Request Identifier (CRI), a **SecureAuth** authentication number that is used only once
- The **SecureAuth** X.509 v3 server information SSL Certificate
- The **SecureAuth** Authentication URL

This information packet is then signed with the user’s private key, by the **SecureAuth** client and then returned to the **SecureAuth** server on validation. **SecureAuth** is insuring the legitimacy of the user by implementing a public/private key pair signing of the user’s **SecureAuth**’s X.509 v3 key pair. In addition, **SecureAuth** is validating that the end user is communicating with the legitimate **SecureAuth** server in two ways:

- The **SecureAuth** client obtains, and signs with user’s private key, the **SecureAuth** X.509 v3 identity certificate
- The **SecureAuth** server signs the **SecureAuth** CRI (Certificate Request Identifier) with the **SecureAuth** private key and then encrypts with the **SecureAuth** user’s public key

This 2-step mechanism to insure that the end user is communicating with the legitimate server provides the authentication with a “defense in depth” procedure. The first mechanism ensures that the SSL termination point for the **SecureAuth** server is the legitimate destination. The second mechanism ensures that the **SecureAuth** authentication packet is not tampered with inside the “corporate cloud”. For example the SSL termination point may exist away from the **SecureAuth** server. For this and for other security reasons, **SecureAuth** has a second defense layer. **SecureAuth** signs the authentication with a one-time **SecureAuth** server private key and then encrypts it with the user’s public key. This mechanism insures that only the **SecureAuth** server is able to validate the **SecureAuth** authentication packet – and not another illegitimate authentication server in the enterprise’s corporate cloud.

The **SecureAuth** X.509 v3 credential is not encrypted with the user’s password. As stated previously, such a method would simply make the credential vulnerable to brute-force decryption attacks. Instead, the **SecureAuth** credential is “fingerprinted” with the information unique to that particular device. In this manner, the credential is NOT valid if removed and is not vulnerable to the off-device brute force attack method. If the end-user requires portability, the end-user simply steps through the **SecureAuth** unique user self-enrollment functionality, (see pages 10-14).

The SecureAuth Second Factor – Encrypted Credentials from the Enterprise Data Store

The above scenario details how the **SecureAuth** X.509 v3 key exchange thwarts man-in-the-middle and other replay attacks with a true “Something you have” authentication credential. This factor is the **SecureAuth** X.509 v3 key pair. *TriCipher* and other products like Arcot® tamper with the user’s password by embedding it into the certificate. Instead **SecureAuth** fingerprints the credential with information that can be only be valid from authentication device.

Once **SecureAuth** has validated the X.509 v3 key exchange, as detailed above, the user is asked to submit a credential that is NOT stored on the **SecureAuth** appliance, but is actually obtained from the enterprise’s data store. Because **SecureAuth** is conducting both the left (client) and right (server) side dialogue in this discussion, **SecureAuth** is able to ensure that this credential is not inspected, stored and replayed in an attack (See Figure 17).

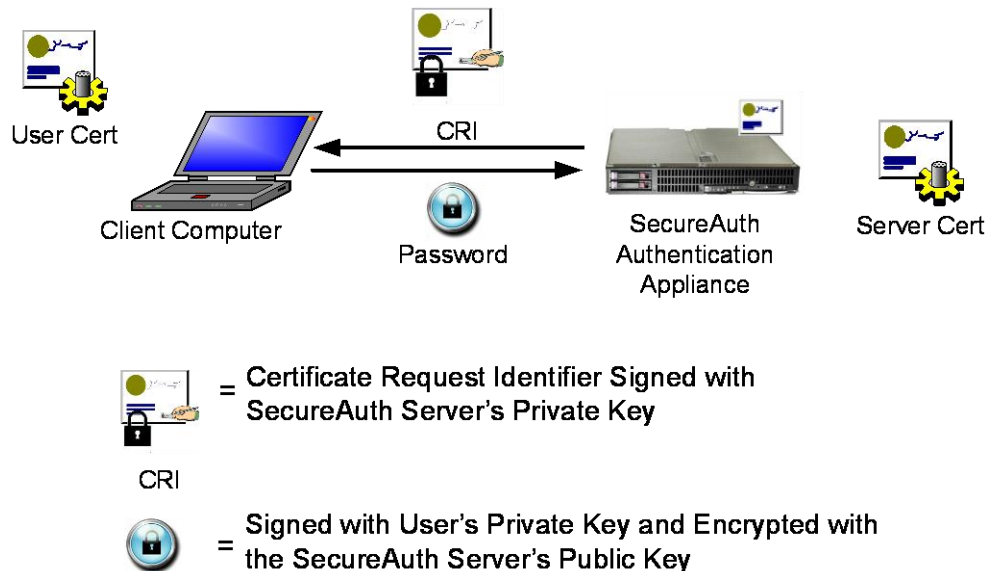


Figure 17: **SecureAuth** ensures that the User’s Credential, the second Factor, is not replayed or altered in transmission.

SecureAuth Server:

- Displays a password prompt, and sends:
 - A signed CRI which is signed with the SecureAuth server’s private key
 - The SecureAuth’s server certificate

The SecureAuth client, on submit of the password:

- Confirms authentication message is from SecureAuth server (The Certificate Request Identifier, CRI, is signed with SecureAuth’s Server private key)
- The SecureAuth client signs the user’s password using the user’s private key
- The SecureAuth client encrypts the user’s password packet with the SecureAuth server’s public key
- The password packet is then transmitted, securely to the SecureAuth server

In the **SecureAuth** second Factor scenario, the **SecureAuth** prompts the end user for his password. This is NOT a **SecureAuth** retained credential. This is the password that the user is utilizing in his other accounts and stored in the enterprise data store (See Figure 17).

SecureAuth does not solely rely on the SSL connection for a safe transmission of this factor. Instead, it uses the **SecureAuth** components that made the first factor, the X.509 v3 certificate authentication valid. The **SecureAuth** JRE client, upon receipt of the user's password, signs the credential with the user's private key and then encrypts the password with the **SecureAuth** server's public key. Once again, this insures that:

- The **SecureAuth** server can be sure the password came from the client's machine
- The User can be assured that only the **SecureAuth** server can decrypt and utilize the password

Summary

The security benefits of using an X.509 v3 certificate for authentication are well documented. In the past, using these certificates required the implementation of PKI. PKI, in turn, comes with limitations on flexibility and usability while imposing onerous expense and complexity.

SecureAuth is a fully automated, simplified and secure solution designed specifically for authentication into protected resources.

The innovations designed into the **SecureAuth** solution result in several advantages when compared with attempting to use a traditional PKI implementation for authentication of users.

By comparison, **SecureAuth** is:

- Easy and quick to deploy
- Simple to maintain
- Far more scalable
- Simple to integrate with applications, federation solutions and network devices
- Easy for end-users to obtain and use credentials
- A more flexible and secure true 2-factor authentication solution

For these reasons, **SecureAuth** should not be considered as another form of PKI. It is a solution that leverages the best security properties X.509 v3 certificates while automating the entire process in a manner which makes the **SecureAuth** solution superior to any other strong authentication product.