eFORCE

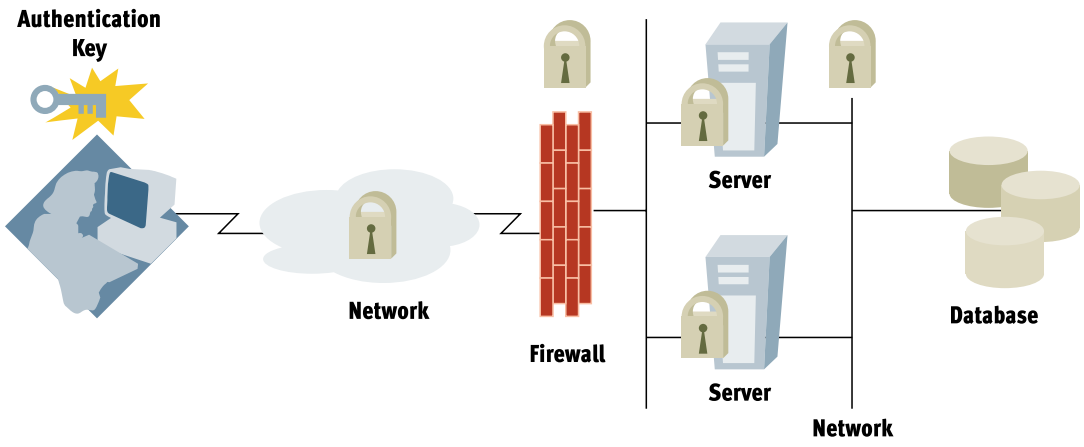
Promoting Web-Enabled eBusiness

Authentication Architectures, Technologies, and Commercial Implementations

For any eBusiness transaction, the authentication process is absolutely fundamental. Without mutual trust in the secure transfer of data, especially that of a personal or financial nature, and reliable establishment of the identity of both consumer and business, eBusiness as we know it could not and would not exist. In order to receive goods, or initiate and complete a transaction, consumers need to ensure that their sensitive information is received only by the relevant and targeted party. On the business side, authentication is necessary to verify the source of the information being transferred in order to protect against fraudulent transactions and theft. This white paper will focus on the criticality of authentication security in the eBusiness environment. It will define and examine various strategies for achieving authentication within an eBusiness system, briefly survey commercial implementations which may be used for authentication in a web environment, and finally, provide a framework for the evaluation of web authentication products.

Whenever a site is exposed to the Internet, security becomes a major issue. The nature and amount of exposure any site has on a public network makes it susceptible to tampering by curious or malicious parties. The degree of risk is of paramount concern at eBusiness sites; those which both provide information to consumers and also allow them to perform financial transactions over a public network. Security exposure in the site translates directly into risk and liability for the company presenting the site to the consumer. As eBusiness broadens to include high ticket items such as vehicles, luxury goods and electronic goods, as well as into the business-to-business market, this risk grows even greater as does the penalty of loss to both business and consumer should a security infringement occur. In the construction of an eBusiness site, authentication assumes a role as important as transaction-processing and will be an issue of paramount importance for eBusiness site owners in the digital economy.

Figure 1. Securing a website involves enforcing multiple levels of control.



Firewalls are used to protect the perimeter of the site to ensure that no malicious or unwanted traffic can gain access to site hosts. Beyond that, each host should be secured and administered with care to ensure that only services that are absolutely required are installed and running, that those services contain the latest security patches, and that those patches have been properly deployed. These two forms of security serve as the 'walls' around the site that offer general site protection. However, the 'doors' through which users enter and perform transactions must also be secured: who may access the site, and the level of access they may have to the site, must also be ensured. This is where authentication plays a critical role. The site must allow users to authenticate themselves in a secure manner. The results of the authentication must also be securely passed to the application that is the business process engine behind the site's front end. A secure hand-off is crucial because the application, upon recognition of who the user is, or what role the user holds, then enforces the appropriate access control. Misidentification of the user by the system, or misrepresentation of the user to the system can result in a security breach and expose both parties to significant risk.

This document outlines the authentication architectures that can be used in such eBusiness environments. The document will describe several authentication technologies as well as present a framework for the evaluation of the available commercial authentication packages. A brief and high-level review of the three security products named will be provided for the framework properties that directly impact the security of an application.

Authentication Architectures

Authentication architectures tend to fall into three broad models: "Something You Know", "Something You Have", or "Something You Are". As with all issues pertaining to security, there is a trade-off between security, cost and usability. In order to determine which models are practical for a given situation, one must examine and be familiar with the relative strengths and weaknesses of each. The next section outlines the relative strengths and weaknesses of these three models, in order to explore which model is practical for a given situation.

"Something You Know"

Something that exists in the user's mind, i.e., passwords or PIN numbers.

"Something You Have"

A device or physical object which one possesses, for example, a token card or digital certificate.

"Something You Are"

A unique, physical characteristic that the user possesses, i.e., voice, finger, or retinal print.

Table 1. The three broad models of Authentication Architecture.

"Something You Know" Architectures

The first model is referred to as the "something-the-user-knows" scheme and is based upon the concept of the user and the host system sharing a secret. It relies upon the user and the authenticator having access to the same key. The many protocols capable of implementing this architecture vary in useful application and robustness as well as in the distribution of the shared secret or the key distribution mechanism. These differences serve as valuable means of comparison between the various protocols.

Most often, this shared secret, or key consists of a password. In the most basic form the password-based architecture is very simple. The user presents the secret, typically a password or PIN, to the host for verification. If the host can confirm the shared secret, the user is permitted access. This model, while adequate in most cases, has two well-known drawbacks. The first is that it hinges on the user maintaining the integrity of the secret. If it is shared with another person or written on a piece of paper taped to the user's workstation, it no longer meets the fundamental requirement of this model – that the secret is known only to the user and the host. The other issue is the "guessability" of the secret. In order to remember the password, a user often chooses a familiar and associable combination or a known English word. Such easy combinations such as birthdays, spouses' names and other determinable passwords lessen the effectiveness of password authentication. Passwords which do not include numbers or symbol keys can be easily broken by 'brute force dictionary attacks' which compare the user's password with a dictionary file to find a match. Once a password has been broken, users' identities and privileges are no longer secured on the system. This form of password or PIN-based authentication is referred to as "weak authentication".

One way of increasing the security of the password-based architecture is by using a one-time password. In this method, the user has a list of passwords, each of which can only be used once. Once used, the password is invalid forever. Even if observed in transit, the password is useless. The susceptibility of the password-based authentication to security breach is relatively high, thus it is most often used in non-commerce oriented websites such as My Yahoo, ZD Net, slashdot.com or monster.com.

"Something You Have" Architectures

The second authentication model is based on 'something-the-user-has', typically a hardware token card or an electronic digital certificate. These certificates and cards are difficult to compromise either by casual observation or through deterministic guessing and are thus much stronger forms of authentication and better ways for a user to prove his or her identity.

The methods of achieving token based authentication have taken great prominence as of late in the industry. The methods of creating public key infrastructures (PKI), and the certificate authorities (CA) that support these token based architectures are very complex. In the token based architecture, security is achieved through mathematical properties rather than a simple shared secret to which a comparison is made. To make this clear, a basic discussion of token- or PKI-based authentication is needed.

Two users, Alice (A) and Bob (B), wish to communicate with each other. Central to this communication is achieving a strong trust that a casual observer cannot fake, an established identity that cannot be falsified, and integrity of the information transferred such that no casual observer can assume or change any of Alice or Bob's information. To achieve this both Alice and Bob generate a pair of encryption keys. Alice generates KA_{pub} and KA_{priv} and Bob generates KB_{pub} and KB_{priv}. Both Alice and Bob publish their public keys (KA_{pub} and KB_{pub}) in such a manner that they can be accessed by anyone who wishes to communicate with either of them. If Alice wishes to send Bob a message (M) so that Bob can verify or authenticate that only Alice could have written and sent it, then she performs the transaction outlined in Figure 2.

Figure 2.

1. Alice generates message M
2. Alice signs the message using her private key KA_{priv}
3. Alice sends the message to Bob
4. Bob uses Alice's public key KA_{pub} to verify that Alice sent it, and no one changed it, and to verify the signature and the message.

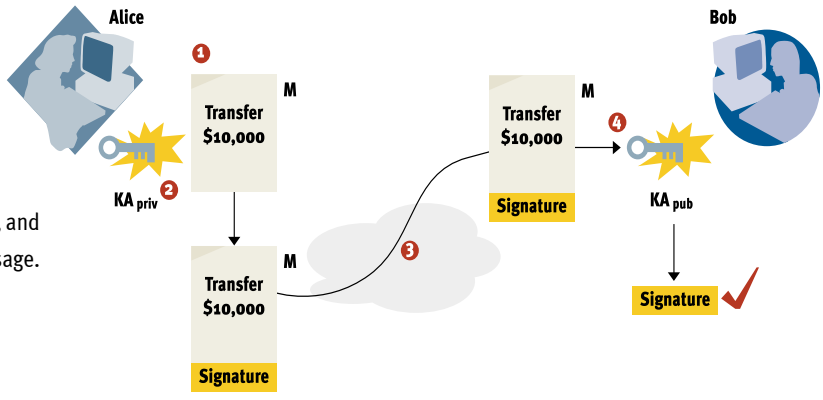
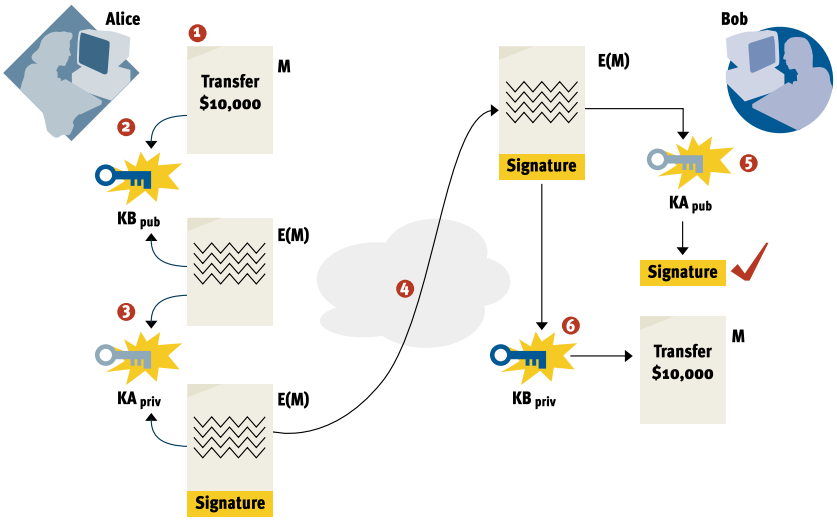


Figure 3. If Alice wishes to send Bob a message (M) that is protected from observation while in transit, and additionally can be verified and authenticated to show that only Alice wrote and sent it, the following transaction applies:

1. Alice generates the message M
2. Alice encrypts the message using Bob's public key KB_{pub}
3. Alice signs the message using her private key KA_{priv}
4. Alice sends the message to Bob
5. Bob uses Alice's public key KA_{pub} to verify the signature and the encrypted message
6. Bob uses his private key KB_{priv} to decrypt and read the message.



Because of the mathematical properties of the public and private keys which are based on large integer factorization (beyond the scope of this paper see Applied Cryptography, *B. Schnier, pp 466-474,* for more details), Bob is able to verify that Alice did sign the message (authentication) and that the message was not changed (integrity). If the message were changed in any manner, Bob would be able to detect this change in the signature of the message.

As described, this process does not protect the message from observation in transit. To achieve this, Alice would have needed to encrypt the message sent to Bob. (See Figure 3.)

Again, because of the mathematical properties of key pairs, a message encrypted under a public key can only be decrypted by use of the associated private key.

The strength of the public key encryption and authentication schemes is rooted in two key components. First and foremost is that the mathematical properties behind the creation of the key pairs is strong and computationally difficult to solve without apriori knowledge of the key pair. The second is that the private key is strictly and safely controlled by the user or entity that created the key pair.

There are several methods of achieving the levels of security that can be derived from public key cryptography. These include both software and hardware token methods. They all rely upon the user "having" something in their possession. What the user has is their private key.

When implemented in hardware, the most common form is a small credit card sized device which holds the key pair for that individual. These cards are often referred to as smart cards. The smart card requires some sort of reader that is capable of pulling the appropriate key information off of the smart card to either encrypt or authenticate the user to the device or service that they are trying to use.

When implemented in software, the most common form is a digital certificate. A digital certificate is simply a software token which contains information identifying the user or entity which is signed by a trusted third party public key. Thus, by presenting the digital certificate, the information stored inside the certificate can be verified or authenticated by means of verifying the signature on the certificate.

However, these approaches are not foolproof either. There is always a chance that a hardware token can be stolen from the user. Similarly, if the user's desktop is compromised, an attacker can gain access to the digital certificates stored on the machine and present it on their behalf. In scenarios like these, this security model is vulnerable to attack. This is due to the fact that the technologies that are currently used to implement the model (e.g. certificates and tokens) do not allow a secured host to verify the identity of a user. The host can only recognize the presence of a mathematically strong object that has been assigned to that user. Authentication based on something a user has is known as "strong authentication".

"Something You Are" Architectures

The third model of authentication is based upon some physical property of the human body, or 'something you are'. This is known as biometric authentication. In this model, the user must present some portion of their human self to be verified for authentication and/or authorization. While this may sound like a page out of a James Bond novel, biometric authentication has made great strides in the past decade.

There are many methods of biometric authentication, including; retinal prints, finger prints, voice prints, and physical measurement just to name a few of the already implemented technologies. There are many factors that make this type of authentication difficult and often impractical. For example, voice timbre and tone changes with mood, health, and age make voice print authentication one of the more difficult technologies. Likewise, retinal scans and physical measurement methods require the user to use often cumbersome and physically uncomfortable devices, and present user acceptance problems. Probably the most convenient and accepted method of biometric authentication is the thumbprint scanner.

However, the devices required to achieve the reliability and accuracy desired are often quite expensive. Inexpensive versions of some of these devices do exist but the old adage of "you get what you pay for" comes to play here very heavily. In general, with the reduction in cost of the devices comes a reduction in accuracy and reliability.

The potential of the biometric authentication has not yet been fully realized. There are devices on the market today which achieve relatively high degrees of accuracy in authentication. Until guaranteed results can be achieved for reasonable costs, biometric authentication will continue to be a high-end choice for maximum security environments and not a mainstream form of authentication. For completeness the biometric authentication was presented here, however, because of the aforementioned liabilities, it will not be presented as a viable alternative for the rest of this paper.

Hybrid Architectures

In order to provide a higher degree of security than any of these models provides by themselves, a hybrid can be implemented. This composite model marries the strong authentication component of a public key system to a user ID and password combination. In such an environment, simply compromising one or the other mechanism still does not allow a rogue user to gain access to a secure system. Both the password and the token or certificate have to be compromised before an attack can successfully occur. This model delivers the strengths of both the prior models, but minimizes the risk of both.

Package Evaluation Framework

In today's fast moving marketplace, the time to build custom authentication systems is often prohibitive. This section of the document seeks to present a framework for the analysis and evaluation of any authentication system. Throughout the framework description, three products — Encommerce's GetAccess, Netgrity's Siteminder and Entegrity's NetCrusader — will be presented as high-level examples of the application for this framework.

Commercial applications have appeal in that they are separated from the application they are protecting. Commercial authentication systems provide a layer of abstraction from the authentication technology is desirable. As authentication technology advances the authentication systems can be upgraded with minimal to no application modification. This allows the maintainers of the web site to stay current with security technology with minimal effort.

Evaluation Framework

With the large number of security products both already on the market today and continually being developed, a comprehensive evaluation of several products would be far beyond the scope of this document. Rather, what is presented here is a framework for the evaluation and analysis of the authentication products. The evaluation criteria are split into two categories, criteria dealing directly with the security principles of the software product and general system criteria that every distributed software archiecture should possess. High level comparisons in each step of the framework using the three products from Encommerce, Netegrity and Entegrity will provide examples of the comparisons to be made. Due to the limited scope and nature of this paper, the example evaluation criteria will be provided for the portions of the framework that apply directly to the security properties of the software product.

Table 1. Framework for the Evaluation and Analysis of Authentication Products.

	Get Access	Net Crusader	SiteMinder
Open Standards	●	●	●
Hybrid Architecture Support	◐	◔	●
State Management and Credential Architecture	●	◕	●
Credential Usage	●	●	◐
Key Storage	◕	◔	●
Ease of Use	●	●	●
Multi-Domain Support	●	●	●
Central & Distributed Administration	●	●	●

Direct Security Evaluation Criteria

1 **Based on open and standards-based technologies:** Due to the evolving nature of web and security technology, it is important to implement a solution that is not proprietary. With an open architecture, the implemented solution can be enhanced with new technologies and standards as they mature. Standards-based technologies such as Kerberos, LDAP, SSL, DES, and RSA provide a solid, peer-reviewed, time-tested set of technologies on which to build an authentication system.

Evaluation: All three products provide use of standard open technologies. There are no proprietary protocols used in any of these products.

2 **Hybrid authentication support:** As discussed previously, it may be desirable to provide a solution that allows some users to authenticate with only user name and password while others may also be required to present a digital certificate or token card authentication. The authentication scheme should allow for such hybrid authentication architectures.

Evaluation:

- **GetAccess**— Does not provide out-of-the-box hybrid authentication functionality. This can be achieved through a custom-built pluggable authentication and authorization module (PAAM).
- **NetCrusader—**Does not provide out-of-the-box functionality. Application APIs could be achieved through programatic control.
- **SiteMinder—**Does provide hybrid authentication mechanisms out of the box. It will allow for BASIC plus certificate authentication.

3 **State management and credential architecture:** Because of the stateless nature of the HTTP protocol, the ability of the site to identify users from HTTP request to HTTP request is critical in any web authentication system. Because a connection between the server and the browser is not maintained, the server must validate each connection to ensure proper security. The credentials used by the system to prove user identity must be built in such a way as to protect against forgery, inspection, and retransmission. Credentials can come in many forms; based on their critical functionality in the authentication system they warrant a more detailed discussion.

Cookies—Because cookies for a particular domain are sent with each HTTP request, they provide a convenient way of presenting authentication credentials to the server. The use of browser cookies to control user authentication is very common. It is essential that any cookie used for such authentication credential be protected. Some of the attacks that can result in a cookie—and thus authentication—becoming compromised include:

- **Sniffing—**Result of sending authentication credentials over insecure channels.
- **Forgery—**Result of poorly constructed cookies.
- **Replaying—**Result of poorly constructed authentication and authorization rules that do not account for timeouts and simultaneous access attempts.

Encoded URLs—The use of URL encoding is an option for some sites that wish to provide a cookie-free environment for their users. In this scheme, after the initial authentication a unique, random, and sufficiently long ID is created and embedded in each URL returned on every page. By analyzing the ID embedded in the URL, the server can revalidate the user requesting a particular page. It is important that these unique URL IDs be truly random, thus imposing the need for a secure, random ID generator whose seeding function is not deterministic. It is also crucial that the ID be from a sufficiently large space in order to prevent brute force attacks on the ID space.

Certificates and Token Cards—The use of digital certificates and/or token cards represents a more secure authentication credential than user names and passwords. But these solutions are not used on each subsequent HTTP request, so the statefull mechanism used to revalidate the authentication upon each request *must* be secure (see Cookies and URLs above).

Evaluation:

- **GetAccess**— Provides for encrypted cookie usage using 56-bit blowfish encryption. Supports the use of passwords, digital certificates, and token cards.
- **NetCrusader—**The cookies that are created are not encrypted but sent only over SSL for confidentiality. This does not protect against workstation level integrity and confidentiality for multi-user machines
- **SiteMinder—**Provides for encrypted cookie session management using RSA RC2 encryption. Credentials are stored in the encrypted cookie in a random order to further protect against observational cryptoanalysis. Passwords, digital certificates, and token cards are supported.

4 **Credential usage:** The application behind the site may wish to reauthenticate the user at any time, providing application level protection of secure data. Thus the credential used to authenticate the used should be available to the application at all times in its original form. In a CORBA environment, the implementation of levels 0, 1 and 2 CORBA security specifications is a key attribute to providing this functionality.

Evaluation:

- **GetAccess**— This product's solution to non-web authentication is its Client Authentication and Authorization Services (CAAS). Based on C++ and Java, the CAAS APIs provide means of authenticating and retrieving authorization information from within an application.
- **NetCrusader—**The non-web authentication is provided through a CORBA/EJB set of APIs capable of providing fine-grained authentication and authorization access to the parameter level of methods. Inter-object communication is achieved through use of Kerberos credentials and secure IIOP.
- **SiteMinder—**Does not natively provide this functionality. SiteMinder provides for a J2EE environment solution which replaces the security context of the application server and provides the application level security and access control desired.

5 **Key storage:** If the system is using a symmetric algorithm for providing authentication such as user name and password, the storage of the keys should be well protected. Additionally, if the authentication system uses symmetric encryption to protect credentials, the keys to this encryption must also be well protected.

Evaluation: All three of the products provide a central storage of the password keys in an RDBMS. And all three systems will support integration with a central LDAP database store of passwords and/or digital certificates.

- **GetAccess—** The GetAccess solution holds the keys to cookie encryption on the web server file system.
- **NetCrusader—**The NetCrusader application does not encrypt the cookies sent to the browser since they are only sent over SSL to the browser.
- **SiteMinder—**This system stores the keys in a protected key store. The key store can be located with the policy server or in a stand-alone LDAP or ODBC.

6 **Ease of use:** The authentication system, while secure, should be straightforward from a usability perspective. It should not deter users from accessing the site due to complexity.

Evaluation: All packages provide for transparent functionality (beyond entry of user name and password) by the user for user name and password-based systems. They also provide only the inherent user experience limitations imposed by certificate and token-based systems.

7 **Multi-domain support:** The ability of an authentication system to support authentication to applications across multiple domains is critical to achieving the single sign-on functionality that many organizations desire. Because many companies use subdomains within their web site architecture or are affiliated with business partners and do not wish to force a reauthentication for each new domain visited, the ability to provide a single mechanism that authenticates across multiple domains is critical.

Evaluation: All packages provide for multi-domain authentication.

8 **Central and distributed administration:** The authentication solution should be as easy to administer as it is to use. The vast majority of exploited security holes are a result of mismanagement and poor administration. Having a system that is easy to administer will prevent the occurrence of such attacks.

Evaluation: All packages provide for distribution of administration duties. Each provides delegation of authority to subadministrators. All packages provide for distribution of administration duties. Each provides delegation of authority to subadministrators allowing for individual departments, groups and organizations to have single user and policy creation, and maintenance, subject to the overall restrictions of the system wide properties.

General System Evaluation Criteria

The evaluation of any distributed software application would not be complete without a review of the general properties of the system that make it useable, robust, and manageable. Because the specifics of these criteria don't relate directly to the security properties of the system, and due to the limited scope of this paper, we will not provide the example evaluation of GetAccess, SiteMinder, and WebCrusader in this section.

1. **Scalable—**As sites are deployed, the anticipated user base will grow dramatically over time. The authentication solution should scale to meet such growth without becoming a bottleneck in the flow of the site.
2. **High Performance—**If the authentication transactions hamper performance and user experience, the purpose of the site will be defeated. The solution should meet acceptable performance requirements.
3. **Replicable—**As sites grow, entire infrastructures are often replicated to international locations for performance and disaster recovery purposes. The authentication mechanism must also be replicated along with the core components of the site. The solution should support multi-site replication of this nature without sacrificing security, performance, or functionality.
4. **Fault Tolerant—**The software architecture implementing the authentication architecture should be fault tolerant and implemented without any single points of failure that would cause downtime for the site or the portions of it that require and/or perform authentication functionality.
5. **Platform Support—**The authentication package should support a wide range of platforms for all components. This will allow for the system to be installed on a platform with which site administrators are familiar and comfortable. While one platform may have a specific advantage or disadvantage with respect to system security, the ability of the site administrators to confidently administer the systems is a key role in the security of any site. The more heterogeneous a system environment is, the more difficult it becomes to stay current on all security and administration practices for that particular platform.

About eFORCE

eFORCE is a leading provider of strategic solutions in all areas of the Enterprise Value Chain — eBusiness, CRM, EAI, Corporate Portals and Business Intelligence. Combining expertise in business architecture, technical architecture, design, deployment and maintenance with its uniquely rigorous, comprehensive eBRIDGE™ implementation methodology (eBusiness Rapid Implementation and Deployment for Global Enterprises), eFORCE delivers production-scale enterprise solutions in Internet time. eFORCE customers include Global 1000 organizations such as Alcatel, Avaya, Bank of America, Charles Schwab, Compaq, DHL, GE Capital, FedEx, Intel, The Hartford, Janus, Johnson Controls, Merrill Lynch, Mitsubishi Motors, Nortel Networks, Visa USA, and Wells Fargo. eFORCE delivers solutions based on best-in-class enabling technologies such as ATG, BEA, Epicentric, E.piphany, HP, Interwoven, iPlanet, Kana, Microsoft, Oracle, Siebel, Sun and Vignette. eFORCE (www.eforceglobal.com) is headquartered in Silicon Valley and has additional Centers of Excellence in North America, Europe and India.

Summary

In summary, the complex and often misunderstood need for authentication services has become even more complex and convoluted with the introduction of the Internet and the need for Web-based solutions. The solutions that have been created still fall into three general architectural models, "something you know", "something you have", and "something you are". To meet these needs, there are several software packages that have been developed to provide this functionality. Determining which commercial application meets the security needs of an organization is often a daunting task. This paper has provided a framework for the evaluation of these packages.