Global Information Assurance Certification Paper

Securing SecurID

Erik Bishop
August, 9 2004
GSEC Practical Assignment
Option 1

Abstract

Today's computer networks hold some of our most important information. Medical, bank and credit card records can all be found on these networks. Most corporate computer networks are accessible through the Internet in some fashion and many employers allow their workers to connect to their networks remotely using VPN technology. Since passwords alone can no longer be considered adequate protection, options such as RSA's SecurID are used to authenticate users through their remote connections. The SecurID system uses a token card to authenticate a user with an authentication server. A number is displayed in a window on the card that changes every 30 to 60 seconds. The user contacts the authentication server, is prompted for a user ID and then a passcode that is a PIN plus the changing number on their SecurID card. Once the user has entered this information correctly and been authenticated by the server, they are passed on to their destination.

Using SecurID on the network perimeter greatly reduces the likelihood that an attacker will be able to circumvent or crack an authentication challenge. However, SecurID too has its weaknesses. Among these are cryptographic attacks aimed at cracking the proprietary SecurID algorithm and attempts to hijack unencrypted sessions between a SecurID user and the authentication server. The formerly mentioned cryptographic attacks are difficult and unlikely, but present nonetheless. On the other hand, session hijacking is not nearly as difficult and may be more effective and less time consuming than trying to analyze and predict the SecurID tokencode system. These attacks rely primarily on two factors: 1) the ability to obtain physical access to SecurID cards and 2) the ability to intercept communications between a user and an authentication server. Tight control of the cards themselves (i.e. inventory and immediate repossession of unused cards) and securing the communication between user and authentication sever will serve to mitigate these threats.

1. Introduction

Since the dawn of time, man has attempted to secure his personal property against the threats of malicious neighbors. In the physical world, this means establishing barriers to deny others access to our valuables. In the logical world, this means creating logical barriers to protect our assets.

The rise of computer based communication and logical resource sharing has created an environment in which the threats we face in protecting our resources expand exponentially, far outpacing our ability to protect against them. This has forced those charged with securing computer systems to add layer after layer to their perimeter's security barriers, trying to outsmart savvy attackers every step of the way. Passwords alone can no longer be relied on by remotely authenticating users to secure the boundaries of the networks that hold our society's most important information.

What can we do to enhance password-based security at the network perimeter? Two-factor authentication, something you have and something you know, was designed as an answer to this question. If an attacker steals what we have, it does not necessarily follow that they will know what we know. Authenticating with both of these articles adds an extra layer to the authentication architecture.

Token-based two-factor authentication is widely used today to protect our computer networks. This method of authentication requires that the user have a device (generally a card) that is synchronized with an authentication server on the network and PIN that is used in concert with the card. The focus of this paper is RSA's SecurID.

2. SecurID

SecurID's most common implementation uses a token card, which is about the height and width of a credit card and about double its thickness. A serial number is engraved on the back of the card that uniquely identifies that token. The card is completely sealed and has a small LCD window on it that displays a constantly changing number. This number is called the passcode or tokencode and it changes about once every minute. The number on the card is synchronized with the tokencode for a particular user on an authentication server on the network.

When accessing the network remotely, a user is usually prompted for their user ID and passcode by a terminal window. Upon first use, the user will be prompted to create a PIN to use in conjunction with the passcode from this point forward. From then on, when the user authenticates, they will use their user ID, PIN and changing passcode to establish a successful connection to the network.

Authentication relies on RSA's proprietary (and secret) algorithm that uses a timestamp generated by a clock on the card to hash the passcode. An administrator with access to the RSA ACE server can control PIN and token

synchronization (as well as initial token card assignment). Some SecurID cards also have a keypad on them that allow the PIN and passcode to be entered directly on the card. These cards are not as commonly encountered as those without keypads.

The SecurID system is generally consistent, however, the card occasionally gets out of synchronization with the authentication server and requires an administrator to log into the ACE server and resynchronize the token. In addition to issues with synchronization, users will forget their PIN. An administrator on the ACE server can also reset this.

When a SecurID card is not used for several months, the time stamp on the card that is used in hashing the passcode can drift far enough away from synchronization with the time stamp on the authentication server that validation becomes impossible. Often, re-synchronizing the card does not work and it will have to be replaced.

The SecurID card is said to have a four-year battery life that renders the card useless at its point of expiration.

3. Vulnerabilities

What are SecurID's vulnerabilities? Like all measures of security, SecurID too has its weaknesses. The first thing to note is that if an attacker has the desire and the time, they will compromise the system. Following are some of the known methods of attacking SecurID. Keeping these attacks in mind will allow us to better protect our SecurID-defended networks and make the attacker's job more difficult.

On the following pages, I will describe specific vulnerabilities to the SecurID system. These vulnerabilities are by no means comprehensive. They are only some of the more commonly discussed weaknesses and should be regarded as the tip of the iceberg in attacking SecurID.

3b. Vanishing Differential

Physical access to a user's SecurID card can only make an attacker's job easier. With the SecurID card in hand, one method an attacker may be able to use is the "Vanishing Differential" attack. Alex Biryukov, Joe Lano, and Bart Preneel put out a technical paper outlining this attack method. Scott Contini and Yiqun Lisa Yin built upon this and showed that the process could be improved.

The attack requires physical access to the SecurID card (preferably several of them). Once the attacker has obtained the card, he will record every passcode

displayed on the SecurID over a period of time (the longer the better). This can be done with a digital camera and OCR software allowing the passcodes to be printed to a text file once photographed. The output is then analyzed for the existence of a "Vanishing Differential." As stated by Scott Contini that is "two consecutive outputs that match another two consecutive outputs later on." With this information, the attacker can obtain the internal key that is used to hash the passcodes. From here, all future passcodes can be predicted and the SecurID process will have effectively been circumvented.

As previously stated, the Vanishing Differential attack requires physical access to the SecurID card, if not several of them. Ensuring that users are educated about this risk and keeping tight control over their SecurID cards thus reduces the likelihood that this attack will be successful

3c. Race Attack

Not all attacks on the SecurID system require physical access to the card. With a small amount of information, an attacker can intercept a user's session with the authentication server and watch as you begin the SecurID authentication process. They can then beat you to authentication so that they gain access while shutting you out.

PeiterZ wrote an essay on this kind of attack. He calls it a "Race Attack." The theory behind it is based on the fact that the SecurID tokencode is a fixed-length string of six numbers. It is known that the last number of the tokencode must be between 0 and 9. So all an attacker needs to do is to be able to monitor the connection between a SecurID user and the authentication server, watching as the tokencode numbers are entered. At this time, the attacker will have setup ten connections to the authentication server and will be mirroring each character that the legitimate user enters. As the second to the last number is entered, the attacker attempts all ten connections with every possible last number. Chances are good that the attacker will beat the user to authentication and gain access to the system.

There are a couple of ways to mitigate this type of threat. One option would be to use line-buffered mode on the local machine if you are using Unix. This will make it so that the tokencode numbers are not transmitted over the network as the user types them in. In Windows, the tokencode can be typed into another application (such as e-mail, text editor, etc.) and then copied and pasted into the SecurID terminal window. And lastly, PeiterZ advises in his essay that the SecurID user be as quick and efficient as possible when entering the tokencode, cutting down the amount of time available for an attacker to do their work.

3d. Session Hijacking

In the Race Attack scenario, the attacker is only able to execute the attack once they have successfully intercepted communications between the user and the authentication server. This sort of “Man in the Middle” activity is not difficult to pull off and can be the entire attack all by itself.
If an attacker intercepts communications on a system that authenticates using SecurID, all he really needs to do is wait until the user has been authenticated and then hijack the session. In this case, the user will have done almost all of the legwork for the attacker.

Using secure communication between the user and the SecurID authentication server mitigates this attack. If the session cannot be viewed in raw data, then it cannot be hijacked as easily.

As an even further simplified version of the above attack, it is possible to intercept communications between the user and authentication server, wait until the tokencode is entered, knock the user off the session and quickly enter the tokencode before it expires. Once again, the line-buffered tokencode entry or secure communication will protect against the possibility of this kind of attack being successful.

3e. Attacking the Backend

All of the attacks mentioned previously in this paper have dealt with the SecurID authentication and have assumed that the attacker actually wants to gain access to whatever our SecurID system is protecting. Although these types of scenarios will be most common, there is another scenario to consider. What if the only thing the attacker wants to do is disrupt communication and prevent users from authenticating?

It is important to recognize that SecurID is a widely used commercial technology and has been implemented in the protection of several high value targets. Biryukov, Lanos and Preneel note in their Cryptanalysis of the Alleged SecurID Hash Function that “more than 12 million employees in more than 8000 companies worldwide use SecurID tokens.” They go on to note that among these “companies” are the NSA, CIA and White House. What would happen if these users were denied entry to their networks?

Perhaps the easiest known way to disrupt SecurID authentication is to DoS the RSA authentication server. SecurID authenticates users on UDP port 5500. Once an attacker has obtained the IP address of the authentication server, he can flood this port and bring the server down.

The best way to deal with these possibilities is to protect and/or monitor the authentication server. This could entail placing it in a DMZ with a sniffer setup to

monitor traffic to and from the server.

3f. Cross-Site Scripting

While SecurID is most commonly encountered protecting the network perimeter, it can be used to secure most anything. It is fairly common to see SecurID protecting web resources. This ability is a standard SecurID feature and works through the web browser and with web servers such as IIS and Apache. When implemented for this purpose, a user will access an authentication page where they will be prompted to enter their user ID, PIN and passcode. The vulnerability lies with the redirector, which does not escape special characters properly. Due to this fact, any request for a URL with special characters in it will cause the server to produce a page containing script that is executed in the user's web browser. With knowledge of this vulnerability, an attacker could trick a legitimate user into entering his passcode information and having that data replayed by the attacker.

This problem was corrected by RSA in an update to their product. Systems are only protected, however, if they are updated. Unfortunately, there are quite likely still outdated implementations of SecurID accessible that have not been updated and are thus unprotected.

This attack brings to the surface a very important detail to consider when implementing SecurID. That is, what systems/software, etc. in SecurID being used with and what are some of those systems' native vulnerabilities? While SecurID may be strong on its own, when it is used on weak systems or systems that are open to exploitation on another level, it becomes moot. The cross-site scripting attack illustrates this point beautifully. SecurID itself is not the problem. The problem is in the integration with HTTP. It is vitally important to ensure that the systems SecurID is to be used with are as secure as possible even without SecurID. SecurID will serve then to provide an extra layer of defense rather than a false impression of security.

3g. Social Engineering

Lastly, the simplest way to defeat any kind of defense at all is probably the use of an attack based on social engineering. Stories have been told of help desk agents that are willing to assign a new SecurID card to an employee who may be having difficulty accessing the network remotely based on the remote user claiming that their card is malfunctioning. Once a new card is assigned, without any way to get the card to the remote user immediately, the help desk agent may even be willing to read the passcode of the newly assigned card to the remote employee and walk them through the process of authenticating. This has happened. This will, unfortunately, happen again.

Although it requires no coaxing of information from a SecurID user, there is another mode of attack that should be listed in the social engineering category. The user ID, PIN and passcode of an authenticating user can be intercepted simply by watching over the user's shoulders as he enters the information.

Consider this scenario: a user sits in a coffee shop and connects to his company's network remotely. He frequents this coffee shop regularly and performs the same routine each visit. A potential has noticed the man's routine and sees an opportunity to take advantage of it. As the man connects to his company's network, the attacker is gathering information by observation. Over time, he has gathered the PIN that the user authenticates with, the user's user ID, the serial number of the SecurID card, the IP address of the company's VPN concentrator and the company's IT help desk phone number. This is more than enough information for the attacker to gain access to the company's system.

The procedure is now as follows. The attacker waits until the man comes into the coffee shop and establishes a connection to the company's VPN concentrator. He is prompted for the usual user ID, PIN and passcode. The attacker, sitting near the legitimate user, enters all of the required information. He enters the current passcode right off of the user's card. The user never even notices. The attacker has now gained access to the network.

In the above attack, The attacker did not have to be very technically savvy. He just had to have a basic understanding of the SecurID process and knowledge of the legitimate user's routine. All of the necessary information to gain unauthorized access to the company's network was offered in plain sight. Had the user been mindful of his public surroundings and kept his SecurID details concealed, this attack could not occur. An even better way to have protected against this attack would have been to avoid connecting to the company's network remotely in a public place.

People are far easier to fool than machines and a social engineering attack doesn't require but the slightest knowledge of how to use a SecurID card and perhaps a telephone. Mitigation lies in administration and security policy. Needless to say, the sort of scenario described here would be impossible if only the help desk agent would have refused to disseminate sensitive information without some sort of procedure to verify the authenticity of the request.

4. Conclusion

Two-factor authentication can greatly add to the strength of our authentication process. RSA's SecurID is probably on of the best and certainly most widely used forms of two-factor authentication. Its vulnerabilities are few and probably are far outweighed by its advantages, but the weaknesses do exist.

When implementing the SecurID authentication system, we need to ensure that the token cards are controlled rigidly. They must be revoked when they are no longer in use and disabled when reported missing. Communications between users and the authentication server should be protected with measures such as encryption. The authentication server should not be accessible to the general public. It should be firewalled and placed in a secure area of the network such as a DMZ. IT staff as well as users must be educated about the ever present threat that our networks face and about the specific ways in which SecurID can be compromised. Good security policy must be implemented that controls to whom (if anyone) sensitive information is disseminated.

When we are realistic about the threats we face and aware of the methods that attackers may take to nullify our defenses, we will increase the likelihood that we get our money's worth from systems such as SecurID.

References

RSA Security, 2004
http://www.rsasecurity.com/

TheFreeDictionary.com, 2004
http://encyclopedia.thefreedictionary.com/SecurID

Scott Contini, “Weaknesses in SecurID FAQ” November, 2003
http://www.crypto-world.com/securid_faq.html

PeiterZ, “Weaknesses in SecurID” Unknown Date
http://www.tux.org/pub/security/secnet/papers/secureid.pdf

Dave Dittrich, “Session Hijack Script” April, 1999
http://staff.washington.edu/dittrich/talks/qsm-sec/script.html

KtKiller, “Inpho on SecurID” May, 2000
http://packetstorm.linuxsecurity.com/groups/9x/articles/

Adam Shostack, “Apparent Weaknesses in the Security Dynamics Client/Server Protocol” October, 1996
http://www.homeport.org/~adam/dimacs.html

Gwendolynn ferch Elydyr, “RSA Aceserver UDP Flood Vulnerablility” July, 2000
http://cert.uni-stuttgart.de/archive/bugtraq/2000/07/msg00187.html

Vin McLellan and Adam Shostack, “Thoughts on the Next Target” June, 1997
http://www.privacy.nb.ca/cryptography/archives/cryptography/html/1997-06/0158.html