



Multi Factor Authentication

The Risks of One-Time Passwords and the Public Key Infrastructure alternative

Introduction

On March 17, 2011, RSA, a division of EMC Corporation, posted an “urgent” warning that a sophisticated intruder could initiate a broad attack against the RSA SecurID one-time password device used by thousands of companies worldwide.¹ Although RSA has been briefing individual customers about possible countermeasures, its executives have not spoken publicly about the nature of the threat. Indeed, even after receiving such briefings, customers and computer security specialists are still puzzled about exactly what the risks might be and what should be done about them.

SPYRUS has not been briefed about this problem, so our knowledge is limited to what has appeared in the press. If press reports are accurate, there is a possibility that the SecurID “root seed file,” a master file of seed keys maintained at RSA, was compromised. If this is the case, there is a chance that virtually all SecurID tokens throughout the world were compromised and must be replaced as quickly as possible.

On the other hand, more recent reports seem to indicate that the attack was a phishing attack directed against certain RSA employees, which allowed the attacker to penetrate RSA’s SecurID server. What might have been compromised as a result of such an attack is not yet widely known. At a high level, what follows is a description of how the RSA SecurID token and similar devices work and why

1. <http://www.nytimes.com/2011/03/19/technology/19secure.html?partner=rss&emc=rss>

The SecurID token and the server share the same seed which means that SecurID is based on a symmetric key algorithm—and that is the main issue

the theft of the master seed key file, or even the seed key file of a specific enterprise, could cause harm

In 2004, RSA updated their SecurID token design to use the newly approved Advanced Encryption Standard (AES).² Every SecurID hardware token receives a unique seed key at manufacturing time that is protected by the tamper-resistant design of the device. RSA software tokens also use a seed key, but the key is injected after the software is installed on a PC or mobile device.

The seed key is used to encrypt the current time of day (64 bits, updated every 30 or 60 seconds), the token serial number (32 bits), and an optional field that may contain other relevant data. The one-time password displayed on the LCD screen of the device or window of the soft token is the result of this calculation. To use the SecurID for authentication, users enter the displayed one-time password along with their user name and static password as part of the authentication process.

The other part of the solution is the RSA authentication server, which sits within the

2. https://groups.google.com/group/comp.security.misc/browse_frm/thread/e00fa564dc6aba5a/1f8529e8df4e02dc?vc=1&hl=en



enterprise's network. The authentication server uses the user name to look up the user's seed key. That seed key is then run through the same algorithm on the SecurID server to validate the information displayed by the token. The number from the token (something you have), along with the user's static password (something you know), provides multi-factor authentication.

Since the time-of-day clock on the token and the time-of-day clock on the authentication server can slip in relationship to each other, the authentication server must sometimes calculate and check both the previous and next result. Time always advances, so the number generated by the token will not repeat for a very long time and is generally considered to be unique. In most cases, this number is far better than a conventional password, because it can be used only once.

Even if the time of day is known, because the algorithm uses the token serial number and other data presumably unknown to an attacker, it is virtually impossible to predict the output of the encryption process. It would be a computationally exhaustive process to try all possible seed keys, and even if that could be done, the authentication server would reject a steady stream of incorrect logon attempts, preventing access to a targeted user's account.

Although no affordable token can be completely invulnerable to penetration, it is relatively unlikely that a token could be penetrated and still continue to function, or that the penetration would go unnoticed by the user. From that perspective, the SecurID token is reasonably solid in its security.

However, since the token and the server share the same seed this means that SecurID is based on a symmetric key algorithm—and that is the main issue.

Symmetric Keys Pose a Problem

The security of any symmetric key algorithm, such as SecurID or even AES, depends on the security of the keys, and the symmetric key must be shared between the originator and the recipient of any communication in order to be useful. Although the SecurID token itself may be secure, the security of

the overall system is dependent on the security of the seed keys. This explains why the possibility that the RSA master seed key file was compromised is such a grave concern.

Even if the RSA master seed key file was not compromised, enterprises should always consider the possibility that the seed key file on their own network might have been compromised. The seed key file must be accessible to the authentication server, and such servers are frequently hacked, and the seed keys must be backed up, presenting yet another opportunity for the seed keys to be compromised.

Public Key Infrastructure: A Secure Alternative

Public Key Infrastructure (PKI) authentication is based on the use of asymmetric keys, where each side of the activity uses one key of a key pair, made up of the "public key" and the "private key." Unlike symmetric key encryption, in asymmetric key encryption, no single key must be present at both locations. The public key can be authenticated to prove that it really is a specific person's or device's public key.

Therefore, only the public keys associated with each authentication device would need to be stored at the authentication server, and even if the master file is stolen it cannot be used to generate authentication requests. In a PKI-based authentication system, the private key in the possession of the user generates the authentication request, while the public key is used to validate the request. And while SecurID can only authenticate the user to the server, PKI authentication also can authenticate the server to the user. Let's see how this works.

Using PKI for Mutual Authentication

A user enters their password to login to their PKI-enabled device and unlock one or more private keys. The password is the first authentication factor: something that you know. Because private keys never leave the device, the device is the second authentication factor: something you have.

The private key can be used to create and digitally sign a "certificate," and the signature can only be

verified using the opposite, or public, key. When connecting to a PKI-enabled server, the server looks for either the certificate (which may contain the user's name or account information) or a reference to the location of the certificate or public key that is used in the next step.

The authentication server now sends a challenge to the device. That challenge could be a random number, a sequence counter, or a time-based challenge. The private key on the device is used to sign the challenge, and the result is then sent back to the authentication server. The authentication server verifies the response by decoding it using the corresponding public key.

Assuming that the signature is properly verified and the response matches what was sent to the user, the user is now authenticated. Additional authentication factors can also be added, including an application password that is checked at the authentication server, biometrics, GPS location, authorized host processors, and other features.

PKI-based authentication has been around for several decades and is widely supported. In particular, Secure Sockets Layer (SSL) and the later Transport Layer Security (TLS) standards have supported PKI-based mutual authentication between a web browser and a server since the inception of those protocols. Just as the server can be authenticated to the user using the https protocol, the most widely used form of PKI, the user can be authenticated back to the server.

Unlike systems that use username and password or similar mechanisms to authenticate the user, hardware-based PKI authentication is not vulnerable to a man-in-the-middle attack, where the browser has been hacked to save and later transmit the user's credentials to the hacker, because the private key is never exposed and the digital signature mechanism takes place in trusted hardware. Because a unique challenge is used in real time, there is no possibility of a replay attack being successful.

SSL and TLS are not the only mechanisms that can provide user authentication. IPSEC can use PKI to provide VPN authentication, and many popular

higher-level applications and protocols can also make use of PKI, including Microsoft Office, Oracle, Adobe Acrobat, and many others.

SPYRUS Was Founded on PKI

Many of the SPYRUS encryption, authentication, and storage products, and in particular the Rosetta smart cards and the Hydra Privacy Card® Digital Attaché, implement public key technology. Each of these devices is capable of generating public/private key pairs using strong, high-assurance, NIST-approved random number generation processes, designed to meet the FIPS 140-2 Level 3 specification. These keys can use legacy RSA and DSA algorithms or the much stronger Elliptic Curve Cryptography (ECC) algorithms.

In addition to generating the keys securely, most SPYRUS products prevent private keys being injected into or extracted from the device. This eliminates the possibility of the private key being compromised by an insider.

Of course, the public key corresponding to the individual private key can be exported, and it can be wrapped in an industry-standard X.509 certificate or saved in a database of such keys when appropriate.

Given that the private keys are thoroughly and carefully protected within the FIPS 140-2 Level 3 boundary of SPYRUS devices, and assuming that the government-approved cryptography is sound, there is no way to work back from the public key to discover the private key.

Summary

The purpose of this white paper was not to make recommendations regarding specific applications, protocols, or authentication servers, but rather to provide information on the use of PKI as an alternative to one-time passwords.

Viable solutions to the risks of one-time passwords do exist, and those solutions can provide considerably higher levels of assurance, with equivalent levels of user convenience and acceptance. In particular, SPYRUS devices using FIPS 140-2 Level 3 public-key-based hardware with strong

Elliptic Curve Cryptography eliminates the need for a symmetric key that is shared between the user's device and the authentication server. This completely eliminates any compromise caused by exposure of the seed keys, either the master seed key file or an individual enterprise's seed keys.

SPYRUS is always available to discuss this topic in more depth. Please see the contact information listed below.

PKI Device Requirements

What follows is a list of functions and standards that you should ask about when evaluating a PKI solution for multi-factor authentication.

Functions

- High-assurance protection for keys, digital IDs, and sensitive data
- Unique serial number for each device
- Advanced random-number generation technology
- Anti-cloning
- Compatible with Microsoft CryptoAPI and Cryptographic API: Next Generation, including PKCS #11. Mac OS supports similar APIs.

Standards Compliance

- SDIO Specification Version 1.10
- SD Physical Layer Specification Version 2.0
- ANSI X9.31 RSA Key Generation
- FIPS PUB 46 Data Encryption Standard
- FIPS PUB 180-2 Secure Hash Algorithm Standard
- SP 800-90 Deterministic Random Bit Generator
- FIPS PUB 186-2 Digital Signature Standard
- FIPS PUB 197 Advanced Encryption Standard
- SP 800-38A Block Modes of Operation
- SP 800-56A Key Establishment Guidelines

Security Certifications

- FIPS 140-2 Level 3 validation

Suite B Cryptography

- Suite B Cryptography is a set of cryptographic algorithms promulgated by the National Security Agency as part of its cryptographic modernization program to serve as an interoperable cryptographic base for both unclassified

information and most classified information, including:

- Elliptic Curve Cryptography (P-256, P-384, P-521)
- ECDH and ECMQV Key Establishment per SP 800-56A
- ECDSA Digital Signature Algorithm
- Concatenation KDF
- RSA 1024 and 2048 digital signature algorithm RSA-1024/2048 key exchange
- DES, two & three-key triple DES with ECB, CBC AES 128/192/256 with ECB, CBC
- SHA-1 and SHA-224/256/384/512 secure hash algorithms with HMAC support

SPYRUS products are proudly designed, engineered, and manufactured in the USA.



Do you know where your security comes from?

For more information about SPYRUS products, visit www.spyrus.com or contact us by email or phone.

Corporate Headquarters

1860 Hartog Drive
San Jose, CA 95131-2203
+1 (408) 392-9131 phone
+1 (408) 392-0319 fax
info@spyrus.com

East Coast Office

+1 (732) 329-6006 phone
+1 (732) 329-6211 fax

Australia Office

Level 7, 333 Adelaide Street
Brisbane QLD 4000, Australia
+61 7 3220-1133 phc
+61 7 3220-2233 fax
www.spyrus.com.au
info@spyrus.com.au



© 2011 SPYRUS, Inc. All rights reserved. Secure Pocket Drive is protected by U.S. Patents 7,757,100, 7,380,140, 6,088,802, and 6,981,149, with other patents pending. Individual Hydra PC products may embody technology protected by one or more of the following patents or patent applications: U.S. Pat. Nos. 6,088,802; 6,003,135; 7,757,100; 7,380,140; 6,981,149; 5,761,305; 5,889,865; 5,896,455; 5,933,504; 5,999,626; 6,122,736; 6,141,420; 6,336,188; 6,487,661; 6,563,928; 6,618,483; U.S. Pat. Appl. Ser. Nos. 12/018,094; 61/300,772; 09/434,247; 09/558,256; 09/942,492; 10/185,735; Can. Pat. Appl. Ser. Nos. 2176972; 2176866; 2202566; 2174261; 2155038; 2174260; E.P. Pat. Appl. Ser. No. 96201322.3; 97106114.8; 96105920.1; 95926348.4; 96105921.9; PCT/US08/51729; Israeli Pat. App. No. 199983; India Pat. Appl. No. 1422/MUMNP/2009. SPYRUS, the SPYRUS logo, Secured by SPYRUS, Hydra Privacy Card, Hydra PC, PocketVault, Digital Attaché, Rosetta, Rosetta Micro, Secure Pocket Drive, SPYCOS, and Security to the Edge are either registered trademarks or trademarks of SPYRUS, Inc., in the U.S. and/or other jurisdictions. All other company, organization, and product names are trademarks of their respective organizations.