PKI - Sham or Salvation?

Chair - Jon David

Lehman Brothers

Panelists -

Padgett Peterson
Lockheed-Martin

Tim Polk
NIST PKI Project

Fred Cohen
Sandia National Laboratories

PKI is touted as the thing that will make the Internet in general, and e-commerce in particular, secure. A barely finite number of companies, most of them new and unheralded, offer PKI "stuff." Just what is PKI, though? Is it as good as the vendors say it is, or is it just another ploy of the marketroids to foist pseudo-solutions on an unsuspecting user base? This session looks at these questions, and answers them.

This session will set forth the intended purposes of PKI, and the related equipment, techniques, protocols, etc. It will detail the benefits it offers, and show how the various components interact to provide the security/authentication/non-repudiation/ etc. associated with it. Since we've all observed that the Internet has not yet reached perfection in security areas, we'll look at PKI realistically, see its shortcomings, and discuss ways -- if any -- to convert what it offers to what it provides.

Attendees, and all others interested in PKI, are urges to visit Carl Ellison's web page, http://world.std.com/~cme, and also the PKI Page, http://www.cert.dfn.de/dfnpca/pki-links.html. In addition, Fred Cohen's web page, http://all.net, contains his 50 Ways to Beat PKI (below) as well as many other worthwhile security writings.

Jon David

Jon David is an officer in the Security Engineering group at Lehman Brothers. With over 30 years in security, he was a pioneer in both computer and network security. Prior to Lehman Brothers, he was Director of Network Security for an ISP, and spent a depressingly long time as a security consultant. He is a frequent author and speaker, and has repeatedly been in the van of security technology. Well past his prime and clearly at the pre-dotage stage of life, he lives off of the knowledge and abilities of friends these days. In an attempt to disguise his street kid tendencies, he did his undergraduate work at Queens College and his graduate work at Columbia University.

A. Padgett Peterson, P.E. CISSP

Employed by elements of the Lockheed Martin Corporation since 1979 first as a digital flight controls specialist, then as senior staff member for embedded airborne computers, and since

1993 as principal engineer - system security with responsibility of corporate chief information security architect.

Has written numerous papers and presented briefings on control systems and security vulnerabilities in both open and classified environments.

One of the “early adopters” of the Internet with secure WAN operations in 1967 and has been telecommuting since 1985.

In addition he developed foundational deployments on commercial systems of digital signatures and encryption (1990), VPNs (1992), and PKI (1994) and has spent a lifetime on the “bleeding edge” of technology.

Mr. Peterson has a Masters degree in System Management and a Bachelor’s in Mechanical/Electrical engineering.

Tim Polk

Tim Polk has been a member of the Computer Security Division at NIST for eleven years, and has been a member of the PKI Project Team for five years. He was co-editor of RFC 2459, the PKIX Certificate and CRL Profile. Tim currently edits several NIST and IETF PKI specifications. Before joining the PKI Team, Tim focused on system security tools and techniques

Fred Cohen

Fred Cohen is one of the most recognized, respected, and requested names in information protection today.

His innovative research, global reach, and unique perspectives have transformed the face of information protection over the last 20 years. He is one of the most widely respected security consultants in the world, a prolific writer, and a leading expert on information assurance, risk management, and critical infrastructure protection. From education, to research, to consulting, he brings the highest quality, the most advanced thinking, and the best strategic analysis in information protection today.

More information: http://all.net/contents/resume.html

50 WAYS TO DEFEAT YOUR PKI AND OTHER CRYPTOSYSTEMS

by Fred Cohen

1. Flood the PKI with fictitious user IDs and names. The net effect is that any typo gets the user one of your pre-positioned keys. You then decrypt their messages and forward them encrypted to the intended recipient.
2. Interrupt traffic every 10 or 20 packets by sending a ‘reset’ packet to the session. Since PKI uses private key encryption for high volume transmissions, it depends heavily on synchronization for its proper operation. Every time you desynchronize the private key system, new public key exchanges have to be made to get a new private key - at the overhead of lots of computer time and exchanged bits. The system will rapidly become unusable.
3. Buy a public key under a phony name and start sending viruses with a signature that is traceable to the fictitious identity. Since key revocation works so poorly today, this will likely be trusted for a long time to come.
4. Buy a new phony public key every few hours and send viruses with them all.
5. Put a Trojan horse in a Word document to send out the users’ private key file. Actually, this was done earlier in 1999, so it’s not an original idea.
6. Put a virus in a Spread Sheet that sends out the users’ private key file. See #5 above.
7. Break into the user’s computer by exploiting some other weakness and forge their private key use.
8. Break into a user’s system and listen to their keystrokes as they type in the password used to reveal their private key (for systems that protect the private key file with some other form of cryptography).
9. Guess the password on the ‘locked’ private key file stolen with one of those techniques above.
10. Use a Trojan horse to disable cryptography without telling the user about it, so that it looks like it is encrypting when it is not.
11. Revoke the key of a user who is still using their key. This will cause them to have to re-register again and again and confuse the whole system over time.
12. Get people to encrypt things you provide them, and use it to get their private key. This was demonstrated to work in the early 1980s.
13. Break into a server that stores public keys and change them to ones you specify.
14. Get into the Internet’s DNS system and change the apparent location of the various key servers on the net. The whole PKI system will break down as no servers can be found anymore to verify anything.
15. Crash a few key servers that form the base of the PKI tree for the users’ you want to defeat and they will be unable to communicate except in plaintext.
16. Observe the time taken to do encryption using peoples’ private keys and analyze the time differentials for different things encrypted to derive the private key bits.
17. Use network time protocol forgeries to make some cryptosystems get desynchronized. The overall effect for many key servers is that they will no longer trust other servers and refuse to communicate with them.

18. As initial key distribution is done, place an attacking machine between the machines exchanging keys and do a man-in-the-middle attack to make each think you are the other. You will now be permanently in the middle of all their communication.
19. Use a van-Eck attack to observe the ‘secret’ messages before they are encrypted.
20. Use a van-Eck attack to observe the ‘secret’ messages after they are decrypted.
21. Use video-viewing to observe the keyboard of the user as they type in their keys.
22. Use video-viewing to observe the keyboard of the user as they type in their messages.
23. Put a fake keyboard adapter between their keyboard and computer and pick up the keys and messages as they are typed.
24. Break into the manufacturer of the PKI system and place a Trojan horse into their system. Nobody will ever find it.
25. Pay the companies to put in a Trojan horse.
26. Force people to use key escrow, and then break the escrow system through legal or extralegal means.
27. Limit key length to only a few hundred bits of key. The government exploited this one for a long time - and still does.
28. Create a false distribution of a PKI system and put it into the infrastructure as a replacement for the original. Then people will download the Trojan version and think they have the real thing. The ‘self-authentication’ will of course claim it is legitimate.
29. Create a clever new crypto-algorithm that has a real subtle backdoor and get lots of people to use it. You will be able to exploit it while everyone thinks it’s secure.
30. Use a parallel processor to break keys of limited length. This has been successful against systems of current common key lengths.
31. Write a computer virus that implements a massively parallel cryptoanalysis engine for breaking private keys. Distribute via the Internet.
32. Modify the pseudo-random number generator used by the cryptosystem to generate keys so that they are relatively easy to guess. Previous systems had poor random number generators so that modification was not required.
33. Find a weakness in the random number generator used today by a popular system and guess lots of real keys in use today. It’s a good bet you can do this if you are smart enough.
34. Cause the systems that generate the pseudo-random numbers to have less randomness than they are assumed to have. Then they will generate poor random numbers without having to do anything special.
35. Generate new private keys for users and plant Trojan horses in their system to have the new keys registered without their knowledge or consent. They you will have a copy of the private key.
36. Generate gobs and gobs of public key traffic and overload all the servers with requests so that the whole system grinds to a halt.
37. Generate millions of public keys (they don’t even have to be legitimate) and push them into key servers to cause them to run out of space and become slow at finding keys.
38. Corrupt the private key distribution packets used to encrypt sessions so that private keys are wrong.
39. Disrupt the private key distribution process so that session keys cannot be distributed.
40. Replay private key distribution in place of interrupted distribution to force old keys to be reused; thus making these sessions easier to break.

41. Store sessions and break them over a long period of time. For most current systems, all of the content will be readable within a few years with parallel processors. For many you can read them today if you spend enough time and effort.
42. Bribe the person who is using the key to get access to the private key.
43. During the electronic distribution of the cryptosystem, corrupt a byte and watch the distribution fail. Do this again and again and the system will fall into disrepute.
44. Create a false update for the cryptosystem and send it out to users. When they load the update, it contains a Trojan horse that defeats the system.
45. Most data exchanged via encryption is stored in plaintext. Get access to the database where they store the information at either end and defeat the value of the cryptosystem in assuring the confidentiality and integrity of the underlying data.
46. Find one of the many flaws in the cryptographic protocol used to do key exchange and private key distribution and exploit it to defeat the cryptosystem. Current cryptographic protocols have been found to have many vulnerabilities.
47. Convince the user to misuse the cryptosystem - for example - to use a weaker cryptosystem because it's faster.
48. Corrupt the browser via a Trojan horse so that it makes it look like the cryptosystem is in use when it is not. For example, make the lock appear to be locked in Netscape.
49. Break into the certificate server and take the master certificate.
    Then you can subvert the whole system without anyone knowing about it.
50. Publish an article on 50 ways to defeat your PKI and cryptosystem, thus eroding public confidence in PKI and making the system less trusted from a public perception perspective. I think I have done that by now.