Hacking PGP

Jon Callas
Black Hat Briefings
Amsterdam
Spring 2005
Overview

- OpenPGP is the most widely-used cryptosystem today
- There ain’t a lock that can’t be picked
- There ain’t a system that can’t be broken

- What is possible? What is not?
- What can we learn from years of experience?
- How do we make things better?
- How do we sanely defend ourselves

- Paranoia is the *unwarranted* fear they’re out to get you
Attackers and Defenders

- There are different axes
  - Good Guys and Bad Guys
  - Attackers and Defenders
- In cryptography, there are only attackers and defenders
  - Some attackers are the good guys
  - Some defenders are the good guys
- Today, we’re concerned only with attacks and defense
Getting the Right Mind-Set

• Typically we think like defenders
  – Look at where we can defend
  – Look at where we can block

• To be a good defender, you need to think like an attacker
  – Imagine what’s possible
  – Imagine what’s out of scope

• Pick your favorite bad guy, and think about how to attack
  – Think about what’s possible with different capabilities, effort levels, threat models
  – If we have X, what can we do?
This Isn’t *Just* Interesting Gossip

- It is important to attack your own system
- It is important to learn how your system is attacked
- It is important to be open about how your system works
- It is important to be open about what your system doesn’t do

- Learn to do this to your own systems
  - You don’t *have* to give a Black Hat talk on it
  - It is good to have a Risks and Threats document at the least
  - I’ve done this for other companies as well.
Assumptions

• I am assuming you know
  – What PGP is
  – Some basic bits of cryptography
    • Crypto scrambles things to make them unreadable
    • There’s such a thing as public and private keys
  – Some basic networking
    • Networks carry data from one computer to another magically
    • Reading this data is easier than we’d like, but harder than some people think
  – Some basic OS security
    • Letting someone write onto your disk is bad

• There are no stupid questions; ask, but I may defer
Terms

• OpenPGP
  – IETF standard for cryptographic data and certificates
  – RFC 2440 -- OpenPGP Formats
  – RFC 3156 -- OpenPGP/MIME

• PGP®
  – PGP Corporation software, implements OpenPGP for messages
  – PGP Disk®
  – PGP AIM encryption

• Other OpenPGP systems
  – Hushmail, GNU Privacy Guard, etc.
Cryptographic Message Structure

- Start with plaintext
- Ordinary data
  - Binary
  - Text

Literal Data
Cryptographic Message Structure

- Compress the Data
- Start hashing if you want to sign

Compressed Data
Data Hash
Cryptographic Message Structure

• Create a signature
Cryptographic Message Structure

- Encrypt the data and signature
  - “Negotiate” a symmetric algorithm
  - Select a session key
  - Start computing a hash for Modification Detection
  - Add MDC packet at the end
Cryptographic Message Structure

- Encrypt the symmetric key to Crypto Recipients
  - All Crypto Recipients get the session key
  - Might be “speculative”
    - Key id is 0
    - Receiving system must try all possible public keys

- Decryption unwinds in reverse order
Public Key Weaknesses

- **RSA**
  - RSA keys can be found if ~1/4 the bits of the private key are leaked
  - A number of attacks (padding attacks, etc.) are based on this
  - Timing attacks, power analysis attacks can leak private key bits

- **DSA**
  - Heavily reliant on random numbers
    - Random number in signature can have covert channels
    - Interesting uses for this, but not relevant to an attack
    - If random number leaks, trivially leaks the private key!
    - This was a key in Defcon ‘04 RootFu
Public Key Weaknesses (cont’d)

- Elgamal
  - Can be used for signatures and encryption, but signatures are fussy, and have been discontinued
- All
  - There is parallelism between factoring and discrete logs
  - If one can be done “easily” then so can the other
  - However, this doesn’t mean we know what the solution is!
  - This may not matter anyway
    - Suppose factoring is found to be polynomial
    - If the polynomial is a big polynomial, it would still be impractical to solve
Factoring Advances

- Directly applies to RSA
  - Mathematically, if RSA is easily factored, there’s an easy discrete log solver
  - No math tells us what it is, just that it exists.
- Adi Shamir estimates that machine to break 1024-bit RSA key in one year can be made for US$10M
  - Easy mitigation -- get a 1025 bit (or larger) key.
  - I’ll be happy to give mine up for a mere US$1M. Such a bargain!
- Bottom line:
  - Even if someone has such a machine(s), are you on the list?
  - If so, get a new key, you’ll be glad you did.
Symmetric Key Weakness

- 8-byte blocks
  - Birthday-attacks after $2^{64}$ message blocks -- $2^{67}$ bytes
  - Only an issue with extended, high-speed transfers
  - This is why AES etc. have 16-byte blocks
- AES
  - Been found to be a large algebraic equation
  - If that equation can be solved, then --- ?
- Encryption Modes
  - CFB mode can be transparently truncated
  - CBC mode (not used in OpenPGP) can be front-truncated
  - Modification Detection Codes (MDC) created to solve this
Symmetric Key Weakness (cont’d)

• Existential Forgeries
  – It is in theory possible to create a message that has the same MDC value as another message
  – Using an HMAC would prevent this
  – Real solution is to sign the message
  – Completely theoretic
  – Easier attack -- just make a new message
    • “I can say I love you just as easily as your SO can.”
Hash Algorithm Weaknesses

- Hash functions falling like flies
  - MD4, MD5, SHA-1, others like RIPE-MD, Haval, etc.
- Going to get worse before it gets better

However:
- Not a single real collision (pre-image collision) has been found even with MD5
- Present attacks of no practical value
- With $2^{69}$ work, I can create two blobs that hash to the same value
- These blobs will be arbitrary? Random?
Cryptographic Strength

- It is easy to forget the power of exponentials
- Every 10 bits is ~1000
- A mole (Avogadro's Number) is about 79 bits
- Are 128-bit keys good enough?
How big is $2^{128}$?

• Imagine a processor the size of a grain of sand
• Assume it can test one key in the amount of time it takes light to cross it
• Make a parallel system by covering the Earth with these to the height of one meter

• How long (on average) does it take to break a 128-bit key?

• Answer: ~1000 years
  – This metaphor courtesy Burt Kaliski
What about Quantum Computers?

- No one knows
- But we think that quantum computers will halve the effective bit size of a key.
- This is why AES has 256-bit keys, as a hedge against quantum computers (or equivalent)
Traffic Analysis

• Encrypted messages stand out
• We can easily see encrypted messages
• Crypto recipients are in plain sight
  – Speculative key ids can hide this
  – Transmission probably makes it obvious, anyway
• If signatures are “outside the envelope” then the signer key id is evident
Anonymized Transmissions

- Even anonymous remailer networks, onion routers, etc. have limits
- If we can see inputs and outputs, they can be correlated
- Fighting correlation introduces latency, and only requires more statistics
Conclusions about Cryptography

- If you find a message *in situ*, there’s not a lot you can do with it
- Key identifiers leak data about recipient
- Hash functions are weakest point, but still ridiculously secure
- Traffic analysis trivially easy, but no eavesdropper can read a message
- None of this is PGP-specific -- everything is affected by these issues
Real-World Example: Accidental RAID on Data

- Locking yourself out
  - Victim did backups of disk -- started playing with a striping array of disks
  - Disks go bad, backup of 15 years of data is encrypted
  - Private key is in the backup

- Situation
  - Without the private key, you’re out of luck
  - Recommended victim look for another backup with the private key in it
  - Fortunately, victim had such a backup from three years past

- Note how he got out of the problem
A Quick Slide on Steganography

- Hide the message in -- something
  - Pictures
  - Sounds
  - Fake spam
- Still subject to traffic analysis correlation
- Severe bandwidth loss
- Works least well against the most obnoxious adversaries
  - An attacker who might just whack you will see stego as proof of guilt
  - Even civilized attackers will see it as admission of being up to no good
What we need is -- the private key

• The way you get at a message is to get the private key
• The private key is encrypted symmetrically with a key derived from the passphrase
• Getting the private key requires getting key data and getting the password
• Hold that thought -- let’s talk a bit more about the network
Oracle-based attacks

- Requires participation of entity that can decrypt message
- Jallad-Katz-Schneier attack
  - Construct damaged version of a message
  - Send to someone who can decrypt
  - Get them to send back erroneously decrypted data
  - Compression, MDC can thwart
- Mister-Zuccherato Attack
  - Construct damaged version of a message
  - Send to someone who can decrypt
  - Get them to report whether quick-check worked
  - ~32K transactions can yield 2 bytes of crypto block
  - Compression makes even less practical
Oracles, cont’d

- With sufficiently stupid recipient, and just the right message, can be a real, effective attack
- Unlikely that humans are sufficiently stupid, but servers could be stupid enough because computers are like that
- Software work-arounds make it not viable

- Bottom line: requires unpatched systems, uncompressed messages, badly built servers

- Protocol discussions in OpenPGP for revisions
Man-in-the-Middle Attack

- Many people mis-characterize MITM attacks
- Here’s what one is:

  - Alice and Bob think they are talking to each other
    - Mallory (in the middle) pretends to be Bob when talking to Alice
    - Mallory pretends to be Alice when talking to Bob
  - This is NOT a simple eavesdropper nor impersonation!
MITM Attacks (more)

• Thwarting MITMs is easy and hard
  – Certificates (including PGP keys)
  – Pre-established relationship (like SSH)
• MITMs are easy to do once, hard to do over the long term
  – If Alice and Bob talk on a channel Mallory doesn’t control, it’s likely to come out
• Not all MITMs are bad!
  – Proxy firewalls, network scan shims (anti-virus, anti-spam, etc.) are all in the middle
• My first PGP Universal paper was called “The Man-in-the-Middle Defense”
• Systems need to differentiate between types of middles (and ends)
Impersonation Attacks

• Mallory pretends to be Bob when talking to Alice, leaving Bob out.
• Phishing, other attacks are impersonation attacks
• Has both technical and psychological components
• Can be very effective in the short run
• Lasts until Alice talks to the real Bob enough
• Alice and Bob can effectively turn the tables on Mallory
Real-World Example: “Voldemort” Incident

- Voldemort tries to find out about a zero-day exploit
- Development team is distributed between Europe (Alice) and US (Bob)
- Voldemort spoofs mail from Alice to Bob
  - Claims to need zero-day fix right away
  - Manages to persuade Alice to “update” PGP to “latest version”
- Voldemort’s insistence on getting information tees off Alice, who phones Bob to chew him out
  - “Look, I told you I’d have the fix by Tuesday, give me a *%$@! Break!”
- Bob says, “Huh?”
- Alice and Bob figure out impersonation, involve the cops
Hacking the Passphrase

- Presumes you have someone’s private key file
  - Snarfed off someone’s computer, computer stolen or seized, insider gets file….
- Now what?
- Start hacking the passphrase
  - Use some cracker
  - Call specialists in this (Access Data, etc.)
  - Major governments have their own clusters to do this
  - Use psychological modeling based upon browser caches, searches of plaintext
  - Can generate over $2^{40}$ hypotheticals per second
Passphrase Hacking (cont’d)

• Things that can help us:
  – Passphrase is in the dictionary
  – 7h3 V1c7im 1z 1336
  – Relates to a hobby or something in the browser cache

• However, this is still a hard task
  – OpenPGP has built-in countermeasures
  – “Iterated and Salted String-To-Key”
  – Hashes passphrase+salt many times to slow down dictionary attacks
  – Cuts rate from millions, billions per second to tens
Physical Access

• “I love cryptography, it tells me what part of the system not to bother attacking” -- Dr Andrew Gross

• Physical Access Is All

• Discussions are now moving away from crypto into systems
• The attacks can get absurdly easy or amazingly clever
Keyloggers

- Hardware or Software
- Hardware keyloggers
  - Might be inserted in serial keyboard cable
  - Might be part of keyboard
  - I know of no USB keylogger
    - This doesn’t mean they don’t exist, and a USB keyboard might be bugged
- Software systems
  - Many spyware systems have them
  - PGP products have some anti-keylogger software
  - Virtualization *could* make this ultimately impossible to detect
Real World Example: Scarfo Case

- Nicodemo Scarfo was a bookie in the US, used PGP
- FBI black-bagged him, installed a keylogger on his system
- Keylogger yielded his passphrase, which was his father’s prison ID #
- We don’t know if it was hardware or software
- Keylogger only worked when he was connected to the Internet

Black bag: spy slang for breaking into a building and stealing things, leaving bugs, cameras, etc..
Other Ways to Get Keys With Physical Access

• Broken random number generator
  – All crypto relies on random numbers for keys, etc.
  – If seeded with (e.g.) time-of-day, makes for easy searches
    • This was a real bug in Netscape Navigator years ago
  – Could be genuinely bogus
    • Suppose it gave out 0-255 -- or was a constant
    • How would you know?

• This is so easy to do I would worry about someone finding it
Other Software Attacks

• Math Games
  – Random fault in RSA signature could release bogus signature that would yield key
  – Malicious blinding or padding could leak information
  – Restricted exponents in discrete logs
  – DSS signatures could release information in covert channel
    • Suppose sig mod 16411 leaked 1 byte of passphrase or key + 6-bit length
  – These software faults would imbed information that an eavesdropper could pick up

• Differential analysis
  – Timing, etc, in non-crypto process.
  – In-machine microphone uses acoustic analysis on computer, keyboard
More Outré and Silly Attacks

• Leak crypto information in other systems things
  – Take 4-bits of data and nibble count. Leak in:
    • TCP/IP sequence numbers
    • Cookies in common web sites

• Think MD5, SHA-1 checksums will help?
  – Hack the ‘md5’ command to return the right value

• While we’re at it, hack the digital signature code to verify what we want

• This is basic rootkit track-covering, just applied to crypto.
Out of Scope But Realistic

- Communications partner compromised, bribed, etc.
- Human intelligence on cleaning staff, repairman, neighbors, self
- Van Eck (TEMPEST)?
- Pinhole cameras in the ceiling, behind a picture, ...
- All depends on threat model
- Don’t forget rubber hose cryptanalysis
Real World Example: The Latvian Incident

- All that is known is from Internet report by purported victim
- Supposedly a whistle-blower in Latvian government
- Snuck out information from government offices
- Information stored on PGP Disk
- Claims that when he was caught, authorities knew things that could only have come from the PGP Disk
- What happened?
  - Is he a troll? Is he wrong? Insane?
  - Was he black bagged? Slipped bogus software? Camera in his house?
  - Merely caught out? Friend, lover suborned?
Back Down To Earth: Mitigation

- Check fingerprints, digital signatures
- Consider your threat model
- Practice good operational security
  - Don’t install things you don’t need
  - Get a laptop, lock it up
  - Store important data encrypted
  - Worry about backups, data warehousing
Tulssing your Software

• Published source is good! That’s why we publish it.
• Published protocols are good
  – Even they end up with issues
  – Secret protocols, source are especially vulnerable to attacker who beats on your system
• External threat assessment, testing is good.
• Hire people to do this.
  – Not even we can rely on beta testing.
• This is like looking under the hood of a car, or visiting the kitchen of a restaurant.
  – All developers should be doing this
  – All users should be insisting on it
Who Built The Software?

• Paradox of open/published source
  – The more available the source is, the easier to hack
  – The more controlled the source, the more the origin is known

• How do you know the verified source is what made the binary?
  – How do you know the waiter didn’t sneeze on your food?

• It is hard to do this retail
  – At PGP, we make desktop sources available, but not installers, etc.
  – For large customers we make full build systems available
  – Ultimately, all developers have to make tradeoffs because there are only 86,400 seconds in a day
The Bottom Line

• What is your threat model?
  – Who is your attacker?
  – What resources do they have?
  – This includes time, money, skills, access to people, software, computers

• What are you defense resources?
  – What can you afford to to defend against?
  – What personal resources can you bring to bear?
  – This also includes time, money, skills, allies….

• PGP was originally designed for activists using BBS systems
Summary

- Cryptography is hard to hack, systems are easy
- The further you are from the victim, the harder it is to hack them
- Weak point is the passphrase
- Information leaks are limited to traffic analysis

- How to hack PGP:
  - Root them
  - Bribe, suborn, compromise someone
  - Black-bag them
  - Steal a private key, break the passphrase
Questions?