Is Encryption Unbreakable?

Steven M. Bellovin

smb@research.att.com

908-582-5886

AT&T Labs Research

Murray Hill, NJ 07974

Is Encryption Unbreakable?

Maybe yes, maybe no — but that's someone else's problem.

- Is the cryptographic protocol correct?
- Is the operating system secure?
- What about the physical container?
- Is the equipment TEMPEST-shielded?
- What are your procedures?

Cryptographic Protocols

- Designing cryptographic protocols is harder than writing programs — and we can't even do the easier task well.
- Many published protocols are insecure. It often takes years to find their flaws.
- Some flaws result from the interaction of the mathematical properties of the underlying cryptosystems with the way the primitives are used.

☞ Though a lot of progress has been made, we have no satisfactory theory of cryptographic protocols.

Is the Operating System Secure?

- Maybe I can't crack the cryptosystem, but I can steal your plaintext.
- On an insecure operating system, one user can steal another's cryptographic credentials.
- Keys themselves can be stolen.
- What about crash recovery files and core dumps?
- Even with secure hardware, there is rarely a trusted path to the crypto hardware — my code can record your Fortezza PIN, and retransmit it to the card.

Operating System Flaws: Examples

- Kerberos stores user session keys in `/tmp` — but what if that directory is mounted remotely?
- Main memory isn't much better; it can be paged out to a network device.
- PCs have no protection whatsoever. (There is published code to steal passwords and credit card numbers.)
- Few modern systems can withstand attacks from logged-in users.
- Java *et al.* provide a vehicle for planting code.

The Physical Container

- Physical access wins — always. (There are reliable reports of ROM being replaced in terminals.)
- An enemy can always add some extra hardware to record things.
- Reverse-engineering often works; boxes are “tamper-resistant”, not “tamper-proof”.
- Induced flaws may be exploitable (i.e., Boneh, Demillo, and Lipton attack on public key systems; Differential Fault Analysis by Biham and Shamir.)

TEMPEST, etc.

- Maybe I can read the plaintext from your screen.
- Improper red/black separation of wiring can be exploited — how many of today's PCs are designed with that in mind?
- This threat is often described in hacker magazines and the like; see, for example, `http://www.thecodex.com/c_tempest.html`.
- How real is the threat? (The writers of these articles don't seem to know much about the inverse square law or S/N ratios. . .)

Procedures

- How do you protect your keying material?
- How do you protect your cryptographic hardware?
- How are cipher systems *used*? (Enigma was cracked largely because of German operational errors.)
- What is the real availability of known or chosen plaintext?
- How is the plaintext protected at either end?
- Can I compromise your facilities?
- Can I bribe the janitor? Can I hire away your people?

In the real world, procedural failings are probably the most serious threat.

Conclusions

- Cipher algorithms may not be the weakest link.
- Enemies are rarely obliging enough to attack at your strong point.
- The name of the game is *information security*, not “can you crack this algorithm?”
- You should worry about your cipher systems — but you should also worry about all the ways around them.