Off-the-Record Communication, or, Why Not to Use PGP

Nikita Borisov
Ian Goldberg
Our Scenario

• Communication privacy is a complicated problem
• Generous assumptions
  – Alice and Bob both know how to use PGP
  – They both know each other’s public keys
  – They don’t want to hide the fact that they talked, just what they talked about
Threat Model

The Internet

Bad Guys

Alice

Bob
Solved Problem

- Alice uses her public key to sign a message
  - Bob should know who he’s talking to
- She then uses Bob’s public key to encrypt it
  - No one other than Bob can read the message
- Bob decrypts it and verifies the signature
- Pretty Good, no?
Threat Model

The Internet

Alice

Bad Guys

Bob
Plot Twist

• Bob’s computer is stolen by “bad guys”
  – Criminals, competitors
  – Subpoenaed by the FBI
• Or just broken into
  – Virus, trojan, spyware, black bag job
• All his key material is recovered
  – Oh no!
Bad guys can...

• Decrypt past messages
• Learn their content
• Learn that Alice sent them
  – And have a mathematical *proof* they can show to anyone else
• How private is that?
What went wrong?

• Bob’s computer got stolen?
• How many of you have never...
  – Left your laptop unattended?
  – Not installed the latest patches?
  – Run software with a remotely exploitable bug?
• What about your parents?
What *Really* Went Wrong

• The software created lots of incriminating records
  – Key material that decrypts data sent over the public Internet
  – Signatures with proofs of who said what

• Alice better watch what she says
  – Her privacy depends on Bob’s actions
Casual Conversations

• Alice and Bob talk in a room
• No one else can hear
  – Unless being recorded
• No one else knows what they say
  – Unless Alice or Bob tell them
• No one can prove what was said
  – Not even Alice or Bob
• These conversations are “off-the-record”
We Like Off-the-Record Conversations

• Legal support for having them
  – Illegal to record conversations without notification
• We can have them over the phone
  – Illegal to tap phone lines
• But what about over the Internet?
Crypto Tools

• We have the tools to do this
  – We’ve just been using the wrong ones
  – (when we’ve been using crypto at all)
• We want *perfect forward secrecy*
• We want *repdudiable authentication*
Perfect Forward Secrecy

• Future key compromises should not reveal past communication
• Use a short-lived encryption key
• Discard it after use
  – Securely erase from memory
• Use long-term keys to help distribute & authenticate the short-lived key
Repdudiable Authentication

• *Do not* want digital signatures
  – Leave non-repudiation for contracts, not conversations

• *Do want* authentication
  – Can’t maintain privacy if attackers can impersonate friends

• Use Message Authentication Codes (MACs)
MAC Operation

Alice

Bob

Data → MAC → MK → MAC

MK → MAC

Data → MAC

=?
No Third-Party Proofs

- Shared key authentication
  - Alice and Bob have same MK
  - MK required to compute MAC
- Bob cannot prove that Alice generated the MAC
  - He could have done it, too
  - Anyone who can verify can also forge
Off-the-Record Protocol

• Rough sketch of protocol
  – Details on our web page
• Assume Alice and Bob know each other’s public keys
  – These keys are long-lived, but we will only use them as a building block
• No forward-secure requirement for authentication
Step 1: Diffie-Hellman

- Alice and Bob pick random $x$, $y$ resp.
- $A \rightarrow B$: $g^x$, $\text{Sign}_{\text{Alice}}(g^x)$
- $B \rightarrow A$: $g^y$, $\text{Sign}_{\text{Bob}}(g^y)$
- $SS = g^{xy}$ a shared secret
- Signatures authenticate the shared secret, not content
Step 2: Message Transmission

- Compute $E_K = \text{Hash}(SS)$, $M_K = \text{Hash}(E_K)$
- $A \rightarrow B$: $\text{Enc}_{E_K}(M)$, $\text{MAC}(\text{Enc}_{E_K}(M), M_K)$
- $Enc$ is symmetric encryption (AES)
- Bob verifies MAC using $M_K$, decrypts $M$ using $E_K$
- Confidentiality and authenticity is assured
Step 3: Re-key

- Alice and Bob pick $x', y'$
- A->B: $g^{x'}$, MAC($g^{x'}$, MK)
- B->A: $g^{y'}$, MAC($g^{y'}$, MK)
- SS' = H($g^{x'y'}$)
- EK' = H(SS'), MK'=H(EK')
- Alice and Bob securely erase SS, x, y, and EK
  - Perfect forward secrecy
Step 4: Publish MK

• Alice and Bob do not need to forget MK
  – They no longer use it for authentication
• In fact, they publish the old MK along with the next message
  – This lets anyone forge messages, but only past ones
  – Provides extra deniability
**IM implementation**

- Instant messaging suited for casual conversations
  - Current security options not satisfactory
- Implemented libotr for secure instant messaging
- Uses:
  - OTR plugin for GAIM (multi-platform IM client for Linux, Windows)
  - Prototype plugin for Adium (OS X IM client based on gaim)
  - Prototype AIM-specific proxy for other clients/platforms
- Toolkit for forging transcripts
  - Any claimed transcript is automatically untrustworthy
Comparison to Other Systems

• gaim-encryption
  – Encryption and authentication
  – No deniability or perfect forward secrecy
  – Like PGP with signatures

• Trillian SecureIM
  – Encryption with perfect forward secrecy
  – No authentication at all

• SILC
  – Completely separate network
  – Share messages (securely) with SILC server, or
  – Pre-shared long-term secret, or
  – Peer-to-peer communication (hard with NATs)
Conclusion

• Current software provides the wrong privacy properties for casual conversations
• We want
  – Perfect forward secrecy
  – Repudiable Authentication
• Use our OTR protocol
  – http://www.cypherpunks.ca/otr/