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

Nikita Borisov
Ian Goldberg
Eric Brewer
Our Scenario

• Communication privacy is a complicated problem

• Simplifying 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
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

Bad Guys

Alice

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
We Like Casual 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 repudiation
Perfect Forward Secrecy

- Use a short-lived encryption key
- Encrypt your data with it
- Discard it after use
  - Securely erase from memory
- Use long-term keys to help distribute & authenticate the short-lived key
Repubdiable 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

Data → MAC

MK

MAC → MK → MAC

Bob

MAC

MK → 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 in the paper
- 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
Step 1: Diffie-Hellman

- Alice and Bob pick random $x$, $y$ resp.
- $A \to B$: $g^x$, $\text{Sign}_{\text{Alice}}(g^x)$
- $B \to A$: $g^y$, $\text{Sign}_{\text{Bob}}(g^y)$
- $SS = g^{xy}$ 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->B: $\text{Enc}_{E_K}(M)$, MAC($\text{Enc}_{E_K}(M)$,MK)
- $Enc$ is symmetric encryption (AES)
- Bob verifies MAC using MK, 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
IM implementation

• Instant messaging suited for casual conversations
  – Current security options not satisfactory
• Implemented OTR plugin for GAIM
  – Multi-platform IM client for Linux, Windows
• Prototype status
  – Help us test it!
What about Email?

- OTR protocol is interactive
  - Requires initial exchange to set up keys
- Can be used for long-term conversations
  - Each round is a message
  - Forward secrecy window days, not minutes
- Can use ring signatures for first interaction
Conclusion

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