Cryptography and the Internet: Where It Is, Where It Isn't, Where it Should Be — and Why It Isn't There...

Steven M. Bellovin
`smb@cs.columbia.edu`
`http://www.cs.columbia.edu/~smb`
Columbia University

Where's the Crypto?

- Any security specialist will tell you that crypto is needed on the Net
- We have not just simple crypto, but many wonderful tricks
- For the most part, though, none of this is used
- Why not?

A Historical Perspective

- The Internet grew from a U.S. military-sponsored project
- For historical reasons — and, in my opinion, sound technical judgment — the NSA preferred outboard crypto hardware
- Traditionally, they used link encryptors, but those are not that useful for the Internet
- In the early 1980s, they started the “Blacker Front End” project for end-to-end encryption and access control
- That grew into SDNS, the Secure Data Network System; SDNS’s SP3 is the ancestor of IPsec
- None of that translated to general-purpose, host-resident, user-available crypto

Why Use Crypto?

- The civilian sector moved more slowly, despite very real threats
- Authentication (attack demonstrated in 1984)
- Sniffing threats (at least since 1993)
- Enable e-commerce (1994-5)

Why Avoid Crypto

- It's complicated
- It's slow
- It's somewhat incompatible with NATs
- Who needs a security blanket?
- The bad guys own the endpoints anyway

Enter E-Commerce and SSL

- Netscape realized early on that people would be scared of buying things on the Internet
- They designed SSL to provide encryption; more importantly, it provides a sense of security.
- Today, every browser has SSL built in
- Some even default to SSLv2 being off...
- But does it help?

Is SSL Useful?

- Almost certainly a technical win — credit card number sniffers are easier to write than password sniffers
- But — the fancy stuff is pretty useless
- 99.999% of users don't know what a certificate is
- Of those who do, most don't verify the certificate details
- Virtually no one knows or cares what CAs their browser trusts (or how those CAs earned that trust)
- Not a good trust chain from the shopping pages to the purchasing pages

How is SSL Used?

- Who knows what a certificate is?

How is SSL Used?

- Who knows what a certificate is?
- Who's ever verified a site's certificate?

How is SSL Used?

- Who knows what a certificate is?
- Who's ever verified a site's certificate?
- Who routinely does it?

How is SSL Used?

- Who knows what a certificate is?
- Who's ever verified a site's certificate?
- Who routinely does it?
- Who has verified the CA's policies?

How is SSL Used?

- Who knows what a certificate is?
- Who's ever verified a site's certificate?
- Who routinely does it?
- Who has verified the CA's policies?
- Who understands and trusts (or even knows) all of the CAs listed in your browser?

The Uses of SSL

- It was extremely important to the development of the web — but much of the benefit was psychological
- The protocol itself has proved useful, since it's easily plugged in in other contexts
- Total secure systems integration remains rare, especially with regard to trust anchors

SSL as an Architectural Mistake

- Unlike IPsec, SSL is application-intrusive
- Unlike SDNS' SP4, SSL doesn't protect TCP headers
- Unlike digitally-signed purchases, SSL does not provide non-repudiation, and requires use of credit card numbers
- Architecturally, it's inferior to all of the alternatives
- It had exactly one advantage: it was deployable. None of the other choices were; they could not have bootstrapped Internet commerce
- Practical utility compensates for a whole host of architectural sins...

Secure Email: S/MIME and PGP

- Two different — and incompatible — ways to protect email
- S/MIME, when available, tends to be reasonably well integrated with MUAs
- But — actual use is very low
- Is the PKI the problem? Do people not perceive the threat?
- PKI software is *very* unpleasant to use; few well-known cases of stolen email to provide motivation
- Beyond that, there's a "network effect" — you can only send secure email to someone else who uses the technology, infrastructure, etc.

IPsec

- Protect everything! Don't touch the applications!
- Host-to-host, host-to-gateway, gateway-to-gateway!
- But — applications can't really take advantage of it, precisely because they haven't been changed
- Host-to-host mode has never really caught on.
- IPsec is used for VPNs, but it's under some pressure there, too

IKE

- The all singing, all dancing key exchange protocol
- Badly specified, poorly implemented, often doesn't interoperate
- Public key mode is the most problematic — PKIs are hard here, too
- But shared secret mode is buggy
- IKEv2 fixes some of these problems, but retains a lot of complexity: it combines a *key exchange protocol* with a *security association management protocol*
- Will IKEv2 ever be adopted?

What's Wrong with IPsec?

- It doesn't interoperate well
- It doesn't interface well to things like RADIUS — the officially preferred approach disagreed with reality, and reality won
- Implementations are very complex to set up

DNSsec

- Major problems with the original design: DNS was not designed to be secured (some of its constructs made life difficult); also, the designers didn't really understand DNS operational practices
- We finally have a spec that appears to be useable
- Well, maybe not — the "authoritative negation" mechanism can be abused to dump the zone; may run afoul of EU privacy law

Lessons from DNSsec

- Design the protocol and the security mechanisms together
- (And design the security mechanisms with provability in mind)
- Pay attention to how the protocol is actually used

Secure Shell

- Nice way to do remote login
- Of course, most of the world doesn't do remote login any more
- Very important, but in niche markets
- Deployable because it requires no infrastructure
- Old wine in new bottles: current target of password-guessing attacks

Password-Guessing

```
Nov 29 00:39:30 machshav sshd[25258]: Illegal user vv from 21
Nov 29 00:39:32 machshav sshd[7307]: Illegal user ww from 217
Nov 29 00:39:32 machshav sshd[20015]: Illegal user ww from 21
Nov 29 00:39:34 machshav sshd[1203]: Illegal user xx from 217
Nov 29 00:39:35 machshav sshd[15441]: Illegal user xx from 21
Nov 29 00:39:36 machshav sshd[8189]: Illegal user yy from 217
Nov 29 00:39:37 machshav sshd[785]: Illegal user yy from 217.
Nov 29 00:39:39 machshav sshd[24449]: Illegal user zz from 21
```

Note that that machine doesn't even *accept* passwords for authentication. . .

Where Crypto Isn't?

- Secure routing
- Cryptographic protection against spam and phishing
- Non-repudiation
- Users. . .

Routing

- Concrete proposals on the table for how to secure OSPF and BGP
- Neither is being used
- The solutions are expensive; worse yet, for BGP it doesn't match operational reality
- People either don't understand the threat, or think that the security costs outweigh the likely losses

Anti-Spam

- Great idea — let's authenticate all email, to get rid of spam
- But — the problem is *authorization*, not *authentication*, and for most users, everyone is *authorized* to send them mail
- Authentication guards against "joe jobs"; that's a minority of the spam
- Besides, most of the spam comes from hacked endpoints; any possible secret key would also be stolen

Anti-Phishing

- What's needed: a strong way to tie email messages back to the original interaction with the financial institution.
- What we have: at best, assertions of "identity" by commercial CAs.
- These are not the same!
- The first phishing attempt I saw was from `paypa1.com`
- If financial institutions start signing their email, we'll see a lot more of that
- There is a cryptographic solution, but is it deployable?

Let Me Enlarge That and Change the Font

paypa1.com

versus

paypal.com

Non-Repudiation

- Do we really need it?
- Real-world signatures don't meet our stringent tests; Xs and printed signatures are perfectly legal
- "Real signatures are strongly bound to the person and weakly bound to the document; digital signatures are weakly bound to the person and strongly bound to the document." (Matt Blaze)
- If the signer's machine has been hacked, the signature means nothing
- Is non-repudiation just a cryptographer's trick?

Non-Use

- Except for SSL-protected credit card number entry, there's very little use of cryptography by the general public
- Some people use VPNs because they have to
- More people use Kerberos without knowing it — it's hidden under the hood of Windows 2000 network authentication
- Virtually no one uses SSL-protected POP3, SMTP, IM, etc.
- Virtually no web traffic is encrypted except for credit card number entry
- Virtually no one uses client-side certificates with SSL
- Why not?

Why Isn't Crypto Used?

- No perceived threat?
- Bad endpoints?
- Too hard to use?
- Operational errors in the design?
- All of the above?

No Perceived Threat

- For most users, eavesdropping isn't a major threat
- It happens, but it's hard to do at scale (though the growth of hot spots may change that)
- The real bad guys prefer to hack the servers
- There are client keystroke loggers — but they evade the crypto

Bad Endpoints

- "Using encryption on the Internet is the equivalent of arranging an armored car to deliver credit card information from someone living in a cardboard box to someone living on a park bench". (Gene Spafford)
- Our host security is incredibly weak
- Most users believe — correctly — that viruses and other malware are bigger threats; crypto won't stop those

Ease of Use

- Much cryptography is fiendishly hard to configure and use
- Too many choices, and too much inherent complexity
- Closed systems can do it invisibly, and do it well
- Users don't notice the crypto with Web browsers, with GSM phones, with Lotus Notes
- Invisible crypto is possible *if* we can deploy the infrastructure
- I assert that ease of use is *the* biggest problem

Operational Errors

- Crypto design must be matched to the operational environment
- The cryptographic trust flow has to mirror the real-world trust flow
- The cryptographic management transactions have to mirror the real world management transactions

Hash Function Follies

- We're about to pay for 10+ years of technical mistakes
- Today's hash functions need to be replaced — they're weaker than they should be
- We tried to design our protocols for hash function agility
- Eric Rescorla and I analyzed five major protocols: S/MIME, SSL, IPsec, DNSsec, and OCSP. Not one got it right.
- Even if we had a new hash function today, we can't deploy it until we fix the protocols and code, and that will take a *minimum* of 5-7 years

Conclusions

- Most of the problems with cryptography are not due to lack of cryptographic science
- We need to do some basic engineering
- We need to do a lot of human factors work
- We need to bind the crypto to reality
- We need to educate users