The Cybersecurity Challenge

Steven M. Bellovin

smb@cs.columbia.edu

http://www.cs.columbia.edu/~smb

The Problem

- "Our first observation is that we are hard pressed to say that cyberspace is more secure than it was 35 years ago"
- "The second observation is that, absent some fresh approach, we are equally hard pressed to say that the situation will materially improve anytime soon"

(Anita Jones and Wm. Wulf)

smb@cs.columbia.edu

It's Not Going to Get Better

- Most security problems are due to buggy code
- Our code is better today than 35 years ago – but the systems we're building are far more complex, and the rate of complexity – and hence bugginess – has increased faster than the code quality
- Even massive efforts, such as the security work Microsoft has put into Windows Vista and Windows 7, hasn't solved the problem

smb@cs.columbia.edu

We're Out of Ideas

- There haven't been any fundamentally new defensive ideas in a long time
- Our basic mechanism is the *wall* – a barrier between good and bad programs, individuals, systems, etc.
- Walls are the easy part – but even they're far from perfect
- The hard part is not the walls, but the gates – the way we permit things to pass through the wall in a controlled fashion

smb@cs.columbia.edu

Seers and Craftspeople

- Many sciences alternate periods of radical change with periods of engineering and minor advances
- In security now, we're in the second phase – but the attackers are stronger than our defenses
- We need radical new ideas

smb@cs.columbia.edu

“Something there is that does not love a wall”

(*Mending* Wall, Robert Frost)

smb@cs.columbia.edu

Firewalls

- We allow many complex things through the firewall
  - Javascript
  - PDF
  - Javascript in PDF
  - More…
- There is not enough sanitization
- Most decent-size companies have many authorized holes – and many more unauthorized ones
- Too many machines – laptops, smartphones, etc. – live both inside and outside the firewall

Operating Systems

- There are too many privileged programs
- Generally, they grant partial privilege to users: they enable some operations that normally would not be permitted, but are acceptable in certain circumstances
  - In other words, they're a form of gate
- The boundary between trusted and untrusted components has been blurred

Applications

- There are many applications (mailers, browsers, PDF viewers, word processors) that are really like operating systems
  - Untrusted input
  - Programmability
  - Resource management
- They're not part of the traditional OS, but failures of their protection schemes can result in user account penetration
- They have their own walls and gates

A Definition

In•san•i•ty (n):

1. Extreme foolishness or irrationality (Mac OS)
2. Doing the same thing over and over again and hoping for a different result (folk wisdom…)

smb@cs.columbia.edu

The Humble Approach

- Our walls *will* fail, and will fail in unpredictable ways
- Our intrusion detection systems are imperfect
- The increased amount of connectivity, through and around firewalls, have rendered them essentially useless

We need a new approach

smb@cs.columbia.edu

The Threat Model

smb@cs.columbia.edu

Threats Have Changed

- The traditional defensive model was implicitly based on the assumption that the good guys had more resources than the bad guys
- That's no longer true – it's often the converse
- There is now much more motivation for attackers

smb@cs.columbia.edu

"Follow the Money"

- Most hacking today is profit-driven
- (Have you noticed how long it's been since a worm shut down the Internet?)
- The market has worked its magic – the attackers now have lots of resources to devote to attacks
- Many of our vulnerable applications were developed on a very tight budget and schedule
- The defenders have to protect everywhere; the attackers get to pick their targets

smb@cs.columbia.edu

Nations

- Most countries have cyberwarfare efforts
- Often, they're the attackers – but the targets are civilian sites running commercial software
- Even governments depend on such software

smb@cs.columbia.edu

New Devices

- We are introducing new devices – and hence new vulnerabilites – without adequate security
- 5 years ago, there was no Facebook
- 5 years ago, there were no iPhones
- 5 years ago, there was no Twitter

What are the security implications of these devices?

smb@cs.columbia.edu

What's Valuable?

- Asymptotically, computers are free
- So are bandwidth and disk space

But…

- People are expensive
- The physical world is valuable
- Data is valuable
- Data is much more valuable in the aggregate; most individual data items aren't that important

smb@cs.columbia.edu

A Research Agenda

smb@cs.columbia.edu

Caveats

- This is a personal vision
- I don't know how to do these things – if I did, it wouldn't be research
- These ideas may ultimately prove just as futile
- But – we haven't mined them out for 35 years

smb@cs.columbia.edu

Themes

- Resilience
- Usability
- Large-scale Systems
- Modes of Thought

smb@cs.columbia.edu

Resilience

smb@cs.columbia.edu

Resilience

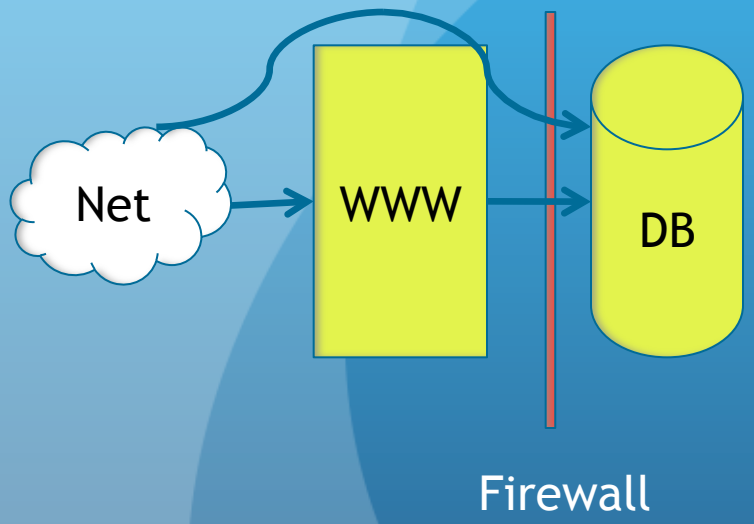
- Today's systems are "brittle" – they can shatter suddenly
- Today, any given subsystem can fall because of a single bug
- "Defense in depth" doesn't work as well as we'd like, because each defensive layer can fail, too
- The security of a system is merely linear in the number of layers – and the constant factor may be arbitrarily small, if the attacker is good enough or lucky enough

Resilient Systems

- A resilient system protects most of its data most of the time
- The rate of data protection failure is low; more precisely, it's *low enough*

smb@cs.columbia.edu

An E-Commerce Site



- Very restricted language from web server to database
  - Simpler language limits bug rate
- Authentication is from the end user to the database
  - *Only active users' accounts are at risk*
- Perhaps even encrypt the database, with the key derived from the users' authenticators

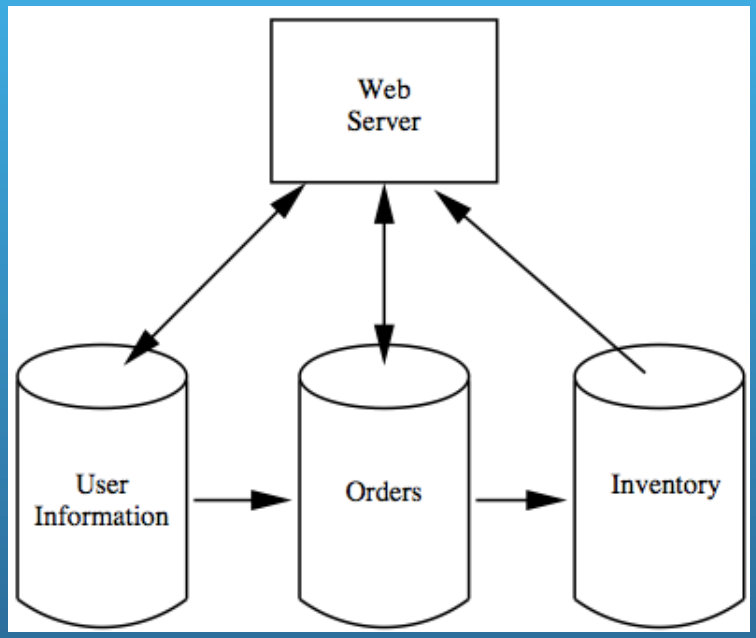
smb@cs.columbia.edu

Web Site Design

- Rate of *data* compromise limited to rate of user activity
- Most users are not active most of the time
- Firewall protects the valuable item – the database – from the outside; the web server is exposed, because it has to be

smb@cs.columbia.edu

Data-Driven Design



- Orders are created by the user database, not the web server
- The order database updates the inventory database
- All write operations by the web server are authenticated by the end-user

smb@cs.columbia.edu

Resilience

- We have restricted the failure modes – no data can be read or (usefully) modified without the authenticator
- Only one small module needs to be correct
- If the IDS works quickly enough, most of the database will remain intact
- We have protected most of the data, most of the time
- (But this design isn't perfect – what are the weak points?)

smb@cs.columbia.edu

Internet-Connected Thermostats

- I recently reviewed the design of an Internet-connected home thermostat
  - Permits remote control of a house's temperature
- The design was not nearly secure enough – an attacker could turn off my heat in the winter, overheat the house in the summer, etc.
- Even if the device had enough crypto and proper authentication, the code might still be buggy (and it probably is…)

smb@cs.columbia.edu

A Better Design

- Have hard-wired limit circuits – never let the temperature in the house get below 5° or above 45°
- Prevent pipes from freezing; prevent plants from dying
- Or – if the limit circuits ever activate, switch control to other hard-wired circuits that keep the house temperature between 10° and 35°, since most people don't want their houses outside that range

smb@cs.columbia.edu

Defining Resilience

- It isn't easy!
- What is a "resilient" car engine computer?
  - (The first cards with microprocessor engine controls had a manual override switch under the hood.)
- What is the analog to temperature limit circuits for an electrical generator, since phase and voltage must be tightly matched to the rest of the grid's?
- Defining the problem is just one of the hard parts

smb@cs.columbia.edu

Usability

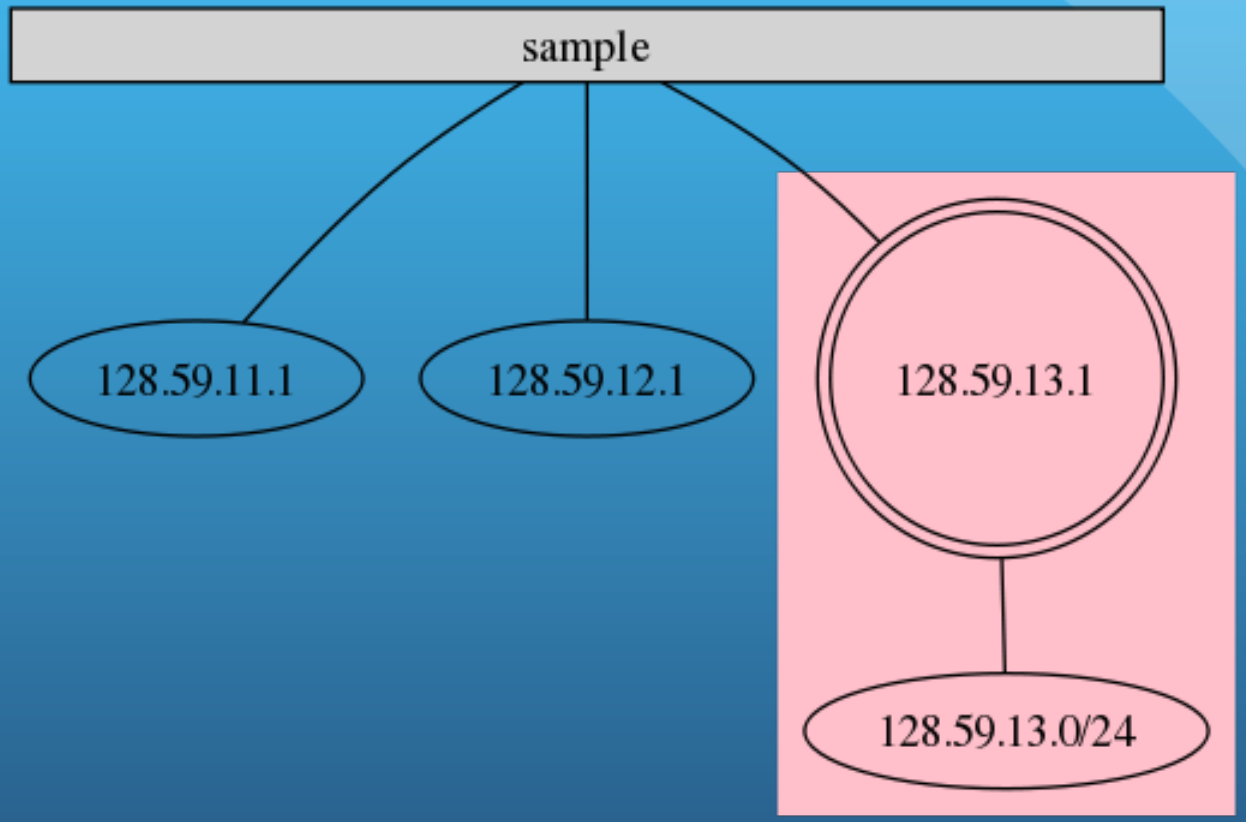
smb@cs.columbia.edu

Usability

- Many of today's security systems are too hard to use
- One reason that phishing happens is that alternatives to reusable passwords are inconvenient
- Even skilled administrators find it almost impossible to configure IPsec VPNs
- Access control policies are incomprehensible

smb@cs.columbia.edu

A VPN Topology



smb@cs.columbia.edu

Configuring it with Simple-IPsec

```
access "direct"    # No triangle routing
type "racoon"      # IPsec implementation
authgen            # Generate certficates automatically
vpn sample {
  nodes "ubuntu" {                  # OS for these nodes
    host 128.59.11.1, 128.59.12.1 # Some remote hosts
    gw 128.59.13.1 {                # Gateway to these nodes
       subnet 128.59.13.0/24        # An entire protected net
    }
  }
}
```

The whole network is configured in one operation; the package-specific files are auto-generated and auto-installed. The graph shown is part of the output.

smb@cs.columbia.edu

Why Is This Better?

- The entire *system* is configured in one operation
- Much of the complexity of IPsec is hidden: there is no way to specify assorted options that never should have existed in the first place
- Other complexity, such as certificate generation, is hidden
- There is exactly one policy decision and one option; everything else is topology or platform+OS

The Access Control Problem

- No one knows how to configure complex access controls, especially in a distributed system
- There are too many interactions, and the effects of any given setting are unclear
  - Which desired operations are now impossible?
  - Which undesired operations remain possible?
- There is no *assurance* that any given selection is correct

smb@cs.columbia.edu

Large-Scale Systems

smb@cs.columbia.edu

Large-Scale Systems

- Today's systems aren't one computer; they're many interconnected systems
- Each is a potential point of vulnerability
- Instead of defense in depth, we have weakness in depth

smb@cs.columbia.edu

Scaling

- We need ways to understand the properties of *systems*
- We need ways for real-world programmers to specify the security properties of the system, just as we did in Simple-IPsec
- We need ways to manage the security settings – including configuration and patch level – of large-scale systems, without very much expensive, buggy human intervention

Modes of Thought

smb@cs.columbia.edu

Modes of Thought

- We don't know how to think about new threats or new services
- More precisely, we approach the questions in an ad hoc fashion, and try to reason by analogy
- Example: what are the consequences of making an iPhone believe a false location?

smb@cs.columbia.edu

Location Threats

- Who is relying on the location?
- Who can spoof it?
- What if it's a car navigation system? A car's speedometer? A geographic access control restriction? An emergency phone call to the police? Location-based advertising?
- The threat will change, depending on the application. How could this be anticipated?

smb@cs.columbia.edu

Extremism

- The usual approach is extremist: either there are no problems, or all new services are banned
- Generally speaking, both are incorrect – but what should replace them?
- Is it possible to have a useful formalism that can describe things that haven't been invented yet?

smb@cs.columbia.edu

Conclusions

smb@cs.columbia.edu

Parting Thoughts

- It is improbable that anyone (including me) will want to give up today's advanced services, let alone all new ones
- But – we are more and more dependent on an increasingly-fragile infrastructure
- My proposed solutions may not be the best, or even the only approaches
- *But we have to try something new!*

smb@cs.columbia.edu

References

- Steven M. Bellovin. Seers and craftspeople. *IEEE Security & Privacy, 5(5)*, September-October 2007.
- Steven M. Bellovin. On the brittleness of software and the infeasibility of security metrics. *IEEE Security & Privacy, 4(4)*, July-August 2006.
- Steven M. Bellovin. Virtual machines, virtual security. *Communications of the ACM, 49(10)*, October 2006.