Lecture 4

Security principles,

military security,

the Orange Book,

and the advent of networks

Orthogonal Security

- Orthogonal security: security mechanisms should be implemented orthogonally to the systems they protect
- Examples:
  - Wrappers to transparently improve system security, e.g. `tcp_wrappers`, `securelib`, sandboxing, etc.
  - Intrusion detection systems
  - IP security, and out-board encryptors
- Advantages:
  - Simpler $\Rightarrow$ higher assurance
  - Applicable to legacy, black-box, untrusted code
  - Can be composed into multiple layers to provide more complete or redundant security

Open Design

"Drive your car to a master mechanic. Tell them that you want a full diagnostic performed on the engine. Tell them that they're to do that, but they can't open the hood to get at it. They will look at you funny."

—Marcus Ranum

- "Security through obscurity" is *dangerous*. This has been known since 1853.
- For security-critical code, you want as many people looking at it as possible
- Remember: the black hats trade info much more readily than the white hats, so security information must be distributed to the white hats (and everyone else) as quickly as possible
- CERT does this *badly*

Open Design, cont.

- Strong vs. weak argument for open design:
  - Weak: Don't *rely* on security through obscurity, because your secrets will leak out eventually
  - Strong: Your system will actually *benefit* from having everyone examine its strength

Open Design, cont.

- But being open doesn't automatically make you secure!
- Firewall-1 was open source for years before anyone actually bothered to look at it

	Closed Systems	Open Systems
Insecure Systems	cellphones, backdoors	Firewall-1, Kerberos, X11
Secure Systems	Military applications	pgp, ssh

A shift of topics

Military security,

the Orange Book,

what worked,

and what didn't

Military security

Military computer security types traditionally worry a lot about

- Confidentiality: the military keeps lots of high-grade secrets.
- High assurance: lives are on the line!
- Insider threats: with millions of insiders, you start to get worried about spies. (Walker, etc.)

The Orange Book: a specification for designing and certifying military-use trusted systems.
Intended to encourage commercial developers to build systems that the military could trust.

Military security, cont.

To understand some of the Orange Book, it helps to understand how military security folks are coming from.

Background:

- Hierarchical authority structures.
- Existing precedents from paper document handling: classification markings, codewords, need-to-know, specialized procedures, etc.
- Serious concerns about personnel security.
- Lots of pre-existing laws and regulations.
- Dedicated and highly secure communications lines.

Orange Book concepts

Some extraordinarily useful concepts from the Orange Book:

- Reference monitors: a mechanism that implements access control, which is
  - tamper-proof,
  - simple enough to trust,
  - and ensures complete mediation
- Trusted computing base (TCB): the smallest subset of the system which must be trusted if it is to be secure
- Assurance: how do you know the TCB does what it claims?
- Multi-level security, mandatory access control, and covert channels.
- Object reuse limitations: objects must be zeroized before reuse.
- Heavy audit logs.
- Trusted path.

Trusted computing base

- often a useful question to ask of any system: can you point to its TCB?
- e.g., in Java, the TCB includes the JVM, the bytecode verifier, the security manager, the classloader, . . .
  - not a good sign for Java's security.

Object reuse

Some spectacular failures:

- MS Word doesn't zero disk blocks before writing to them, so sending a Word document to someone else may leak confidential data. Oops!
- Kerberos 4 also has object reuse vulnerabilities: in some cases, it will send a packet containing data from an unsanitized buffer.

Trusted path

- Trusted path means when you see a “login:” prompt, knowing it’s really the trusted login program and not a Trojan horse surrogate.
  - e.g. in NT, pressing CTRL-ALT-DEL gives you a guaranteed-good login prompt
- More generally, “trusted path” is about how you bootstrap trust.

Mandatory access control

- Motivation: subjects are not entirely trusted.
- Intended to prevent attacks by insiders (both untrusted users and untrusted code).
- Based on pre-existing classification procedures:

  > No person may have access to classified information unless: (a) that person has been determined to be trustworthy, i.e., granted a personnel security clearance – MANDATORY, and (b) access is necessary for the performance of official duties, i.e., determined to have a need-to-know – DISCRETIONARY.

- All data is tagged with a classification level (UNCLASSIFIED, SECRET, TOP SECRET, . . .) and optionally a list of codewords (NOFORN, ULTRA, . . .).
- Codewords allow for compartmentalization (need-to-know).
- Information cannot flow “down” without proper authorization (de-classification by an authorized security officer).

The lattice model

Mathematically, the MAC policy above lends itself to a lattice model.

- Let C be the set of classifications, W the set of codewords.
- A label l is $l = (l_c, l_w)$ with $l_c \in C$, $l_w \subset W$.
- You get the partial ordering $\leq$ on tags by

$$l \leq l' \qquad \Leftrightarrow \qquad l_c \leq l_{c'} \text{ and } l_w \subset l_{w'}$$

- Informal policy: you may not read "up" or write "down"
- Formal rules for access by a subject with tag l to an object with tag l':

  read allowed only if $l \geq l'$; write allowed only if $l \leq l'$.

- Only trusted software may violate these rules.

MAC and integrity

A serious problem with mandatory access control:

- The policy above assures only confidentiality, not integrity.
- Note that it allows an unclassified user to tamper with even the most sensitive data!
- Can obviously ensure integrity with a lattice model, too
  - but the rules are the dual: you may not write up or read down
- And if you try for both mandatory confidentiality and integrity, the access rules degenerate to the trivial condition that $l = l'$, which is too inflexible.
- Moral: you can't have your cake and eat it too.

What goes up may never come down. . .

- In practice, the tendency is for objects to migrate up the lattice, and soon everything takes on the highest classification level possible.
- A few simple thought experiments:
  - Imagine an Excel spreadsheet containing only unclassified data. You add a single classified field somewhere near the end. Then you try to print the first page, which contains only unclassified fields, to an unclassified printer. Will the system let you?
  - When sending messages from low to high, how do you handle ACKs?
  - What's the security label on the program counter register?

What goes up may never come down. . . (cont)

- You end up needing to build “trusted” data pumps which can de-classify data.
  - requires either human intervention or solving a nearly AI-complete problem
  - and thus is a very difficult problem
- This is yet another serious problem with MAC.

Covert channels

A covert channel is a stealthy way to send information from high to low:

- e.g. by CPU load: to send a “1” (“0”), spin-loop (sleep) for a second; to receive, measure CPU load for a second.
- All sorts of crazy covert channels: monitor disk head timings, amount of free memory, etc.
- Theoretically, the *non-interference* property is sufficient to prevent covert channels.
- In practice, you end up isolating all programs with different security labels, and eliminating all resource sharing/multiplexing from your OS.
  - Once you’re done, does it really deserve to be called an OS anymore?
- In real life, eliminating covert channels is mind-bogglingly difficult.

This is yet another serious problem with MAC.

Assurance

- Arguably the most important criterion of all!
  - After all, a false sense of security is worse than no security at all
- Note that the Orange Book ties assurance into the certification levels; its more recent European cousin makes assurance and features orthogonal.

Problems with the Orange Book

It doesn’t fit the commercial world, and in some ways it is just plain silly.

Problems with the Orange Book

- Certification takes many years.

  In a market living on Internet time scales where time-to-market dominates, this is unthinkably deadly.

- Certification is linked to hardware configuration.

  You end up with obsolete hardware, too.

- Any fix, patch, version rev., or change in hardware configuration requires a re-certification.

  Oops, there goes another year. . .

- If you want to buy a B1-system, you end up with some 386 running 10-year old software, and it'll cost you $100k for the privilege.

Problems with the Orange Book

- The security policy is all wrong.
- The Orange Book is heavily concerned about confidentiality and insider threats.
- But most commercial enterprises trust all insiders.
  - Anyhow, defending against employees leaking information by covert channels is silly, when they can easily walk out with floppies in their briefcase.
- And availability and integrity controls are more important than confidentiality in many cases.

Problems with the Orange Book

- The threat model is all wrong.
- In a B1 firewall, you turn off all web access:
  - just in case some spy on a secret network does something totally goofy like going to the web page

    `http://www.kgb.gov.su/the/attack/begins/at/midnight`
  - so you end up telling your users to *go home* and use an AOL account.
- Anyhow, worrying about covert channels is just plain silly:
  - nobody in their right mind is going to try to steal information from disk head timings when they can have Aldridge Ames for $100k!

Credits: Marcus Ranum

Problems with the Orange Book

Interestingly, the Orange Book applies only to non-networked computers. This is very explicit.

> Once you plug that C2-certified NT box into a network, it loses its certification!

Results?

Software glitches leave Navy Smart Ship dead in the water

July 13, 1998: In September 1997, the Yorktown suffered a systems failure during maneuvers off the coast of Cape Charles, VA., apparently as a result of the failure to prevent a divide by zero in a Windows NT application. The zero seems to have been an erroneous data item that was manually entered. Atlantic Fleet officials said the ship was dead in the water for about 2 hours and 45 minutes.

To first order, noone buys multilevel certified systems anymore; everyone is using a Windoze 95 or NT box.

Lessons

- Worrying about covert channels is the wrong mindset.
  - Much safer: simply never let untrusted programs get access to confidential information.
- MAC probably doesn't help much against untrusted insiders.
  - But it can help stop Trojan horses.

Networks

So what happens when you introduce networks anyway?

Answer: the whole game changes.

Networks and centralized control

- Networks eliminate all hope of centralized control.
  - unless you magically ensure that your network is secure, and only trusted machines are connected to it
  - and even if you can, it is totally fragile.
- This dooms all hope for multi-level systems or mandatory access control.
  - Do you label each packet with its classification level?

Networks and the threat model

In practice, just about the only reasonable threat model is to assume that the network is totally under the control of the adversary.

Why?

- Hard to ensure physical security of network links.
- Multiplexed access means the wire may go right into the enemy's doorstep.
- Every piece of infrastructure (routers, nameservers, . . .) introduces single points of failure.
- When every computer on the net has access to every other, scaling is your enemy.
  - This makes it especially critical that all public services be perfect.

Networks

Networks require new tools:

- Techniques for dealing with scale.
- Techniques for secure communications, and for managing trust.

The answer to the latter is cryptography. More next week…