ECS 235B, Lecture 16

February 13, 2019

Endpoint Protection

- Control how TCP state is stored
  - When SYN received, entry in queue of pending connections created
    - Remains until an ACK received or time-out
    - In first case, entry moved to different queue
    - In second case, entry made available for next SYN
  - In SYN flood, queue is always full
    - So, assure legitimate connections space in queue to some level of probability
    - Two approaches: SYN cookies or adaptive time-outs

SYN Cache

- Space allocated for each pending connection
  - But much less than for a full connection
- How it works on FreeBSD
  - On initialization, hash table (*syncache*) created
  - When SYN packet arrives, system generates hash from header and uses that to determine which bucket to store enough information to be able to send SYN/ACK on the pending connection (and does so)
    - If bucket full, oldest element dropped
  - If peer returns ACK, entry removed and connection created
  - If peer returns RST, entry removed
  - If no response, repeat fixed number of times; if no responses, remove entry

SYN Cookies

- Source keeps state
- How it works
  - When SYN arrives, generate number (*syncookie*) from header data and random data; use as ACK sequence number in SYN/ACK packet
    - Random data changes periodically
  - When reply ACK arrives, recompute syncookie from information in header
- FreeBSD uses this technique when pending connection cannot be inserted into syncache

Adaptive Time-Out

- Change time-out time as space available for pending connections decreases
- Example: modified SunOS kernel
  - Time-out period shortened from 75 to 15 sec
  - Formula for queueing pending connections changed:
    - Process allows up to b pending connections on port
    - a number of completed connections but awaiting process
    - p total number of pending connections
    - c tunable parameter
    - Whenever $a + p > cb$, drop current SYN message

Other Flooding Attacks

- These use *reflectors* (typically, infrastructure systems) to augment traffic, creating flooding
    - Attacker need only send small amount of traffic; reflectors create the rest
    - Called *amplification attack*
- Hides origin of attack, which appears to come from reflectors

Smurf Attack

- Relies on router forwarding ICMP packets to all hosts on network
- Attacker sends ICMP packet to router with destination address set to broadcast address of network
- Router sends copy of packet to each host on network
  - If attacker sends steady stream of packets, has the effect of sending that stream to all hosts on network
- Example of an *amplification attack*

DNS Amplification Attack

- Uses DNS resolvers that are configured to accept queries from any host rather than only hosts on their own network
- Attacker sends packet with source address set to that of target
  - Packet has query that causes DNS resolver to send large amount of information to target
  - Example: zone transfer query is a small query, but typically sends large amount of data to target, typically in multiple packets, each larger than a query packet

Pulse Denial of Service Attack

- Like flooding, but packets sent in pulses
  - May only degrade target's performance, but that may be enough of a denial of service
- Induces 3 anomalies in traffic to target
  - Ratio of incoming TCP packets to outgoing ACKs increases dramatically
    - Rate of incoming packets much higher than system can send ACKs
  - When attacker reduces number of packets to target, number of ACKS drop
  - Distribution of incoming packet interarrival time will be anomalous
- Vanguard detection scheme uses these 3 anomalies to detect pulse denial-of-service attack

Key Points

- Availability in security context deals with malicious denial of service
- Models of denial of service have waiting time policy and user agreement as key components
- Network denial-of-service attacks, and countermeasures, instantiate these models
- Amplification attacks usually hide origin of attacks, and enable flooding by an attacker that sends a relatively small number of packets

Overview

- Chinese Wall Model
    - Focuses on conflict of interest
- CISS Policy
    - Combines integrity and confidentiality
- ORCON
    - Combines mandatory, discretionary access controls
- RBAC
    - Base controls on job function

Chinese Wall Model

Problem:

- Tony advises American Bank about investments
- He is asked to advise Toyland Bank about investments

- Conflict of interest to accept, because his advice for either bank would affect his advice to the other bank

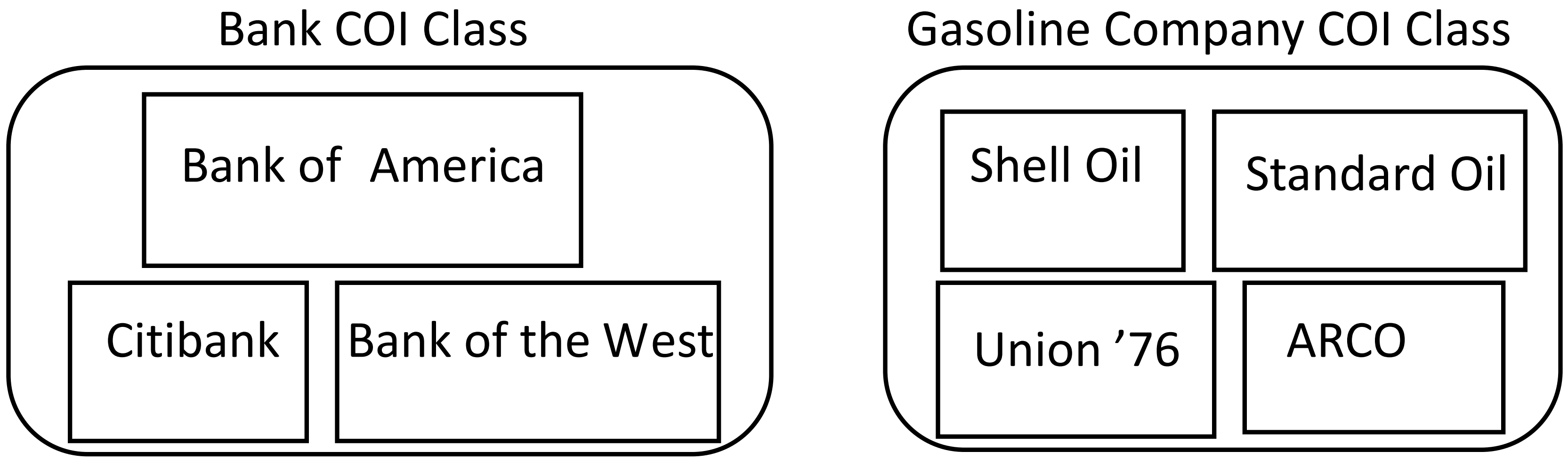
Organization

- Organize entities into “conflict of interest” classes
- Control subject accesses to each class
- Control writing to all classes to ensure information is not passed along in violation of rules
- Allow sanitized data to be viewed by everyone

Definitions

- *Objects*: items of information related to a company
- *Company dataset* (CD): contains objects related to a single company
  - Written *CD*(*O*)
- *Conflict of interest class* (COI): contains datasets of companies in competition
  - Written *COI*(*O*)
  - Assume: each object belongs to exactly one *COI* class

Example


Bank COI Class
Bank of America
Citibank
Bank of the West
Gasoline Company COI Class
Shell Oil
Standard Oil
Union ’76
ARCO

Temporal Element

- If Anthony reads any CD in a COI, he can *never* read another CD in that COI
  - Possible that information learned earlier may allow him to make decisions later
  - Let *PR*(*S*) be set of objects that *S* has already read

CW-Simple Security Condition

- *s* can read *o* iff either condition holds:
  1. There is an o' such that *s* has accessed o' and $CD(o') = CD(o)$
     - Meaning *s* has read something in *o*'s dataset
  2. For all $o' \in O$, $o' \in PR(s) \Rightarrow COI(o') \neq COI(o)$
     - Meaning *s* has not read any objects in *o*'s conflict of interest class
- Ignores sanitized data (see below)
- Initially, $PR(s) = \varnothing$, so initial read request granted

Sanitization

- Public information may belong to a CD
  - As is publicly available, no conflicts of interest arise
  - So, should not affect ability of analysts to read
  - Typically, all sensitive data removed from such information before it is released publicly (called *sanitization*)
- Add third condition to CW-Simple Security Condition:

  3. *o* is a sanitized object

Writing

- Anthony, Susan work in same trading house
- Anthony can read Bank 1’s CD, Gas’ CD
- Susan can read Bank 2’s CD, Gas’ CD
- If Anthony could write to Gas’ CD, Susan can read it
  - Hence, indirectly, she can read information from Bank 1’s CD, a clear conflict of interest

CW-*-Property

- *s* can write to *o* iff both of the following hold:
  1. The CW-simple security condition permits *s* to read *o*; and
  2. For all *unsanitized* objects *o*′, if *s* can read *o*′, then *CD*(*o*′) = *CD*(*o*)
- Says that *s* can write to an object if all the (unsanitized) objects it can read are in the same dataset

Formalism

- Goal: figure out how information flows around system
- S set of subjects, O set of objects, $L = C \times D$ set of labels
- l_1: $O \rightarrow C$ maps objects to their COI classes
- l_2: $O \rightarrow D$ maps objects to their CDs
- $H(s, o)$ true iff s has *or had* read access to o
- $R(s, o)$: s's request to read o

Axioms

- Axiom 8-1. For all $o, o' \in O$, if $l_2(o) = l_2(o')$, then $l_1(o) = l_1(o')$
  - CDs do not span COIs.
- Axiom 8-2. $s \in S$ can read $o \in O$ iff, for all $o' \in O$ such that $H(s, o')$, either $l_1(o') \neq l_1(o)$ or $l_2(o') = l_2(o)$
  - *s* can read *o* iff *o* is either in a different COI than every other *o′* that *s* has read, or in the same CD as *o*.

More Axioms

- Axiom 8-3. $\neg H(s, o)$ for all $s \in S$ and $o \in O$ is an initially secure state
  - Description of the initial state, assumed secure
- Axiom 8-4. If for some $s \in S$ and for all $o \in O$, $\neg H(s, o)$, then any request $R(s, o)$ is granted
  - If *s* has read no object, it can read any object

Which Objects Can Be Read?

Theorem 8-1: Suppose $s \in S$ has read $o \in O$. If s can read $o' \in O$, $o' \neq o$, then $l_1(o') \neq l_1(o)$ or $l_2(o') = l_2(o)$.

- Says *s* can read only the objects in a single CD within any COI

Proof

Assume false. Then

$H(s, o) \wedge H(s, o') \wedge l_1(o') = l_1(o) \wedge l_2(o') \neq l_2(o)$

Assume *s* read *o* first. Then $H(s, o)$ when *s* read *o*, so by Axiom 8-2, $l_1(o') \neq l_1(o)$ or $l_2(o') = l_2(o)$, so

$(l_1(o') \neq l_1(\mathrm{o}) \vee l_2(o') = l_2(o)) \wedge (l_1(o') = l_1(o) \wedge l_2(o') \neq l_2(o))$

Rearranging terms,

$(l_1(o') \neq l_1(o) \wedge l_2(o') \neq l_2(o) \wedge l_1(o') = l_1(o)) \vee (l_2(o') = l_2(o) \wedge l_2(o') \neq l_2(o) \wedge l_1(o') = l_1(o))$

which is obviously false, contradiction.

Lemma

Lemma 8-2: Suppose a subject $s \in S$ can read an object $o \in O$. Then s can read no o' for which $l_1(o') = l_1(o)$ and $l_2(o') \neq l_2(o)$.

- So a subject can access at most one CD in each COI class
- Sketch of proof: Initial case follows from Axioms 8-3, 8-4. If $o' \neq o$, theorem immediately gives lemma.

COIs and Subjects

Theorem 8-2: Let $c \in C$. Suppose there are n objects $o_i \in O$, $1 \leq i \leq n$, such that $l_1(o_i) = c$ for $1 \leq i \leq n$, and $l_2(o_i) \neq l_2(o_j)$, for $1 \leq i, j \leq n$, $i \neq j$. Then for all such o, there is an $s \in S$ that can read o iff $n \leq |S|$.

- If a COI has n CDs, you need at least n subjects to access every object
- Proof sketch: If s can read o, it cannot read any o' in another CD in that COI (Axiom 8-2). As there are n such CDs, there must be at least n subjects to meet the conditions of the theorem.

Sanitized Data

- $v(o)$: sanitized version of object o
  - For purposes of analysis, place them all in a special CD in a COI containing no other CDs
- Axiom 8-5. $l_1(o) = \mathrm{l}_1(v(o))$ iff $l_2(o) = l_2(v(o))$

Which Objects Can Be Written?

Axiom 8-6. $s \in S$ can write to $o \in O$ iff the following hold simultaneously

1. $H(s, o)$
2. There is no $o' \in O$ with $H(s, o')$, $l_2(o) \neq l_2(o')$, $l_2(o) \neq l_2(v(o))$, $l_2(o') = l_2(v(o))$.

- Allow writing iff information cannot leak from one subject to another through a mailbox
- Note handling for sanitized objects

How Information Flows

Definition: information may flow from o to o' if there is a subject such that $H(s, o)$ and $H(s, o')$.

- Intuition: if s can read 2 objects, it can act on that knowledge; so information flows between the objects through the nexus of the subject
- Write the information flow between o and o' as (o, o')

Key Result

Theorem 8-3: Set of all information flows is

$$\{ (o, o') \mid o \in O \land o' \in O \land l_2(o) = l_2(o') \lor l_2(o) = l_2(v(o)) \}$$

Sketch of proof: Definition gives set of flows:

$$F = \{(o, o') \mid o \in O \land o' \in O \land \exists\, s \in S \text{ such that } H(s, o) \land H(s, o'))\}$$

Axiom 8-6 excludes the following flows:

$$X = \{ (o, o') \mid o \in O \land o' \in O \land l_2(o) \neq l_2(o') \land l_2(o) \neq l_2(v(o)) \}$$

So, letting *F** be transitive closure of *F*,

$$F^* - X = \{(o, o') \mid o \in O \land o' \in O \land \neg(l_2(o) \neq l_2(o') \land l_2(o) \neq l_2(v(o))) \}$$

which is equivalent to the claim.

Aggressive Chinese Wall Model

- Assumption of Chinese Wall Model: COI classes are actually related to business, and those are partitions
  - Continuing bank and oil company example, the latter may invest in some companies, placing them in competition with banks
  - One bank may only handle savings, and another a brokerage house, so they are not in competition
- More formally: Chinese Wall model assumes the elements of *O* can be partitioned into COIs, and thence into CDs
  - Define *CIR* to be the conflict of interest relation induced by a COI
  - For $o, o' \in O$, if o, o' are in the same COI, then $(o, o') \in CIR$

The Problem

- Not true in practice!
  - That is, in practice *CIR* does not partition the objects, and so not an equivalence class
  - Example: a company is not in conflict with itself, so $(o, o) \notin CIR$
  - Example: company *c* has its own private savings unit; *b* bank that does both savings and investments; oil company *g* does investments. So $(c, b) \in CIR$ and $(b, g) \in CIR$, but clearly $(c, g) \notin CIR$

The Solution

- Generalize *CIR* to define COIs not based on business classes, so *GCIR* is the reflexive, transitive closure of *CIR*
- To create it:
  - For all $o \in O$, add (o, o) to *CIR*
  - Take the transitive closure of this
- Then (o, o') ∈ *GICR* iff there is an indirect information flow path between o and o'
  - Recall (o, o') ∈ *CIR* iff there is a direct information flow path between o, o'
- Now replace the COIs induced by *CIR* with generalized COIs induced by *GCIR*

Compare to Bell-LaPadula

- Fundamentally different
  - CW has no security labels, Bell-LaPadula does
  - CW has notion of past accesses, Bell-LaPadula does not
- Bell-LaPadula can capture state at any time
  - Each (COI, CD) pair gets security category
  - Two clearances, *S* (sanitized) and *U* (unsanitized)
    - *S dom U*
  - Subjects assigned clearance for compartments without multiple categories corresponding to CDs in same COI class

Compare to Bell-LaPadula

- Bell-LaPadula cannot track changes over time
  - Susan becomes ill, Anna needs to take over
    - C-W history lets Anna know if she can
    - No way for Bell-LaPadula to capture this
- Access constraints change over time
  - Initially, subjects in C-W can read any object
  - Bell-LaPadula constrains set of objects that a subject can access
    - Can’t clear all subjects for all categories, because this violates CW-simple security condition

Compare to Clark-Wilson

- Clark-Wilson Model covers integrity, so consider only access control aspects
- If “subjects” and “processes” are interchangeable, a single person could use multiple processes to violate CW-simple security condition
  - Would still comply with Clark-Wilson Model
- If “subject” is a specific person and includes all processes the subject executes, then consistent with Clark-Wilson Model

Clinical Information Systems Security Policy

- Intended for medical records
  - Conflict of interest not critical problem
  - Patient confidentiality, authentication of records and annotators, and integrity are
- Entities:
  - Patient: subject of medical records (or agent)
  - Personal health information: data about patient’s health or treatment enabling identification of patient
  - Clinician: health-care professional with access to personal health information while doing job

Assumptions and Principles

- Assumes health information involves 1 person at a time
    - Not always true; OB/GYN involves father as well as mother
- Principles derived from medical ethics of various societies, and from practicing clinicians

Access

- Principle 1: Each medical record has an access control list naming the individuals or groups who may read and append information to the record. The system must restrict access to those identified on the access control list.
    - Idea is that clinicians need access, but no-one else. Auditors get access to copies, so they cannot alter records

Access

- Principle 2: One of the clinicians on the access control list must have the right to add other clinicians to the access control list.
  - Called the *responsible clinician*

Access

- Principle 3: The responsible clinician must notify the patient of the names on the access control list whenever the patient’s medical record is opened. Except for situations given in statutes, or in cases of emergency, the responsible clinician must obtain the patient’s consent.
    - Patient must consent to all treatment, and must know of violations of security

Access

- Principle 4: The name of the clinician, the date, and the time of the access of a medical record must be recorded. Similar information must be kept for deletions.
  - This is for auditing. Don’t delete information; update it (last part is for deletion of records after death, for example, or deletion of information when required by statute). Record information about all accesses.

Creation

- Principle: A clinician may open a record, with the clinician and the patient on the access control list. If a record is opened as a result of a referral, the referring clinician may also be on the access control list.
  - Creating clinician needs access, and patient should get it. If created from a referral, referring clinician needs access to get results of referral.