6.915 Computer and Network Security **Fall Term, 1995**

Lecture 13–14–15 : November 24–26–31

Lecturer: Rosario Gennaro, Ron Rivest Scribe: Matt Fredette, Rosario Gennaro

1 Lecture 13

Lecure 13 covered the following topics:

- Trojan Horses
- Mandatory Access Control
- The Bell-LaPadula confidentiality model.

A good survey of the material can be found in Handout 22.

2 Lecture 14: Outline

In this lecture we will discuss the following:

- Term paper topics
- Integrity - what does it mean? How do you achieve it?
- `tripwire`
- The Biba model
- The Clark-Wilson (CW) model

3 Term paper topics

The due date: The term paper topic is due November 7.

There are many possibilities for topics for a term paper, the only strict criteria are that the topic be interesting and somehow related to security. A term paper could be a study of the literature available on a particular subject, or even a design project involving the creation of an actual protocol. The term paper should be around fifteen pages long. Collaboration is okay as long as the ambition of the topic and the paper's length expand too. Perhaps these term papers will be handouts for the class next year?

As far as actually finding a topic, one way to start would be to take the outline of the course, pick one concept and write a paper discussing it in more depth than we would otherwise in class. It is important to coordinate with Professor Rivest if you intend to do your paper on a topic that will be covered later in the course.

Other good sources of term paper ideas are conferences' papers. Many can be found in hardcopy in the LCS reading room or Professor Rivest's office. The Web is also a good place to look, especially starting from `http://theory.lcs.mit.edu/ rivest`. Good conferences include EUROCRYPT and CRYPTO. Also look at the ACM's security series or other recent proceedings for hot topics.

In the case of a design project of an actual protocol, it will probably be very useful to look at some of the available crypto toolkits for primitives to use. Just to give some ideas right now, good papers could be written on:

- cryptographic hashes
- certification (especially public-key)
- key distribution
- electronic commerce
- scrutinization of a standards committee - "rip it apart" (for example, the committee for IPv6)
- a solution to the White House need for digital signatures on the emails that they send (re: Rivest's email)
- security issues involving some "hot" software packages like Tcl/Tk, Java, Perl

- election protocols
- tempest attacks
- issues (psychological/social, technological) with the MIT card reader. some buildings get them, others don't. Big brother concerns. Half electronics, half physical. Discuss hacking of the system.
- re: Wall Street Journal article about company's software notifying the company over a network about the software's (mis?)use. There are concerns about another institution, like the CIA, doing the same thing. Related to ability of Microsoft's Windows '95 to automatically notify Microsoft of the collection of software installed on your machine. Is this legal? Is it moral? There is the all-to-common problem of something being "legal until there is a law passed otherwise".

The bottom line is that there are term paper projects everywhere. The only hard and fast requirement is that a term paper be interesting and security related.

4 Integrity

Integrity is usually identified as being one of the major goals of security, with confidentiality being an example of another. But what does integrity really mean? Some brainstorming yields that "having" integrity probably should mean:

- preventing all unauthorized changes.
- logging all changes (for the purposes of undoing or reverting)
- having the ability to verify the origin/source of data. this is related the "preventing all unauthorized changes."
- maybe tolerating some changes as long as the functionality (for programs, "expressivity" for data?) is preserved. maybe called "no undesirable changes".
- doing double-entry bookkeeping, for verifying internal consistency.

The only thing that is clear is that integrity is a "fuzzy" word whose definition is not formalized. This is probably a good candidate for a term paper.

5 `tripwire`

`tripwire` is a program that calculates a checksum for every file, directory, and symbolic link in (some subset of) a filesystem. Usually the checksums are cryptographic hashes. All of the checksums are stored in the Tripwire database, with the idea being that you run it once, then run it later to determine what has changed since the last time.

How does this involve integrity? Well, it certainly tells you when it may have been violated, by telling you what files have changed. An integrity violation has probably occured if the system administrator runs `tripwire` and notices that `/bin/login` has been changed. But `tripwire` doesn't give you any help in learning *how* integrity has been violated; it can't necessarily tell you when, how, or by whom a file was changed. It can't tell you if the functionality or "expressivity" of an executable or data file has changed - other mechanism is needed to do that. `tripwire` basically can only warn, but in the case of viruses, having a warning of possible infection is exactly where detection and cleansing must begin.

6 The Biba model

A discussion of the Biba model can be found in the Amaroso text, the "runner-up" text for this course. The Amaroso text also happens to have an excellent biblography, including "The 25 Greatest Works in Computer Security", which would be a great place to dig for a term paper topic.

The Biba model deals with subject and objects, much like the Bella-LaPudula (BLP) model. Subjects and objects have *integrity ratings*, synonymous with the *classifications* from the BLP model. The Biba model compares the integrity ratings associated with files, processes, and people.

What should integrity ratings be? How should they be assigned? Can there be a universal collection of such ratings? There is the problem of deciding how to assign integrity ratings to things like your local version of Emacs, executables that you create yourself, a program you receive over email from a friend, or one you grab from a Usenet newsgroup. This is a *level of trust* problem, and is somewhat related to the sentiment of making sure that "you always pack your own parachute".

Other questions are, How should ratings evolve? Does a file's need for high integrity increase with use? There is the question of judging a file's need for integrity at all

- for example, a file may need high integrity, but not high secrecy, or it may be a different combination of higher secrecy, but lower integrity.

The actual description of the Biba model can be found in the readings handout for this lecture. It looks like the Bella-LaPadula model upside-down: in the BLP model, reading an object at a lower secrecy level is allowed, but reading an object at a higher secrecy level is disallowed, while in the Biba model, reading an object in a lower integrity level is disallowed, but reading an object at a higher integrity level is allowed. Similarly, in the BLP model, writes to a higher or equal secrecy level are the only writes allowed, while the BLP model allows writes to only the same or lower level of integrity. Summarized, the BLP model disallows "read-ups" and "write-downs", while the Biba model disallows "read-downs" and "write-ups".

Because the allowed operations in the two models are opposite, there is the real problem of creating a system that implements both models simultaneously. Does being at a high BLP secrecy level imply a high integrity level anyways? When something is at a high BLP level, that usually means a smaller number of people can modify it - does this mean that it will have a higher integrity, by virtue of the fact that it probably *won't* be changing much? At second thought, it probably doesn't mean this, since there may be tools can (accidentally) be used that may not preserve integrity in the eyes of most. When the BLP and Biba models are used in any combination, there is the real problem of making sense and maintaining consistency of the two different sets of ideals.

7 The Clark-Wilson model

The Clark-Wilson (CW) model is another dealing with the issue of integrity. A detailed description of it can be found in the readings for this lecture. The model tries to ensure the integrity of data objects, called "Constrained Data Items" (CDIs). The model also speaks of "Integrity Verification Procedures" (IVPs) and "Transformation Procedures" (TPs). IVPs are meant to verify that CDIs are in a valid state, that their integrity has not been violated. TPs are the only procedures that are allowed to modify CDIs, or to take arbitrary user input and create new CDIs. IVPs and TPs are specifically certified to verify and transform, respectively, as the creators of the system see as necessary.

In the CW model, the users of the system are specifically constrained, by a form of access list, to modifying a stated set of CDIs, using a certain set of TPs. The model also calls for authentication of users, and that modifications be logged, for the purposes of creating an audit trail, and so that changes might be undone.

8 Handouts

The handouts for this lecture included the midterm exam, and a packet containing two chapters from the E. Amaroso book *Fundamentals of Computer Security Technology*, from Prentice Hall: Chapter 9, about the Bella-LaPadula model, and Chapter 12, about the Biba integrity model. The packet also contained the paper about the Clark-Wilson model, *A Comparison of Commercial and Military Computer Security Policies*

9 Lecture 15

The topics for lecture 15 was fingerprinting of digital data. The material is covered in Handout 25.