This week we will be learning about security policies, multilevel security, monitoring systems, and nuclear command and control. In Week 2, we learned the general rules of matrix access control. This week we will study several specific systems whose security is enhanced by nicely defined security policies. To build a secure system, a top-down approach will typically take the form of: *threat model* - *security policy* - *security mechanisms*. Each of these steps is equally important. However, in practice, the second step is often misinterpreted or completely overlooked. This week's seminar will enable us to understand the importance of a well-defined security policy for a project. The study of security policy can be broken down into three main areas: the security policy model, security target and protection profile. This lecture offers an overview of several security policy models. See the textbook for more on security targets and protection profiles.

Multilevel Security and Security Policy Models

Bell LaPadula (BLP) Model

Before we describe the BLP model, let's first take a look at the model used by military systems. In this model, each subject and object is associated with one element of a fully ordered set. For example, the ordered set might be: {open < confidential < secret < top secret}. Objects associated with one particular element may only be used by those subjects whose associated elements are as high as, or higher than, those of the objects. The application of this scheme to governmentally classified data is straightforward. However, there may also be applications in commercial environments.

The above model however can easily be defeated by attacks such as a Trojan horse - a useful, or apparently useful, program or command procedure containing hidden code that, when invoked, performs some unwanted or harmful function. Trojan horse programs can be used to accomplish functions indirectly that an unauthorized user could not accomplish directly. For example, a Trojan horse placed in a program used by a top-secret subject may write top-secret objects to an open object. Despite the fact that there was no evidence so far that anyone had yet inflicted a serious Trojan horse attack on military systems, such an attack was obviously seen as tempting by some - and people became concerned about the possibility. In the light of these concerns, David Bell and Len LaPadula formulated the Bell LaPadula (BLP) model in 1973. This is also known as the multi-level security model. In addition to the 'no read up' property of the above model used by military systems, BLP model also enforces the property of 'no write down'. Formally, the Bell-LaPadula model enforces two properties:

- The *simple security property*: no process (subject) may read data (object) at a higher level. This is known as *no read up* (NRU);
- The **-property*. No process (subject) may write data (object) to a lower level. This is also known as *no write down* (NWD).

The *-property was Bell and LaPadula's critical innovation. Without it, a multi-level security system cannot protect itself against malicious codes, a virus or Trojan. Many people

though feel that the ability to write below the process's security level is a necessary function - for example placing data that is not sensitive, but contained in a sensitive document, into a less sensitive file so that it can be made available to people who need to see it. United States Department of Defense (DoD) experts thought it is more important to protect systems against the threat of de facto downgrading and therefore felt the model had to preclude it. All work sponsored by the National Computer Security Center (NCSC) has since employed this model.

Note: the name for the *-property was originally a 'placeholder' - pending giving it a more formal title. But it has never been replaced.

We turn now to discuss briefly the relationship between the BLP model and access controls. Generally, there are two kinds of access controls.

Mandatory access control, which, according to DoD Trusted Computer System Evaluation Criteria is 'a means of restricting access to objects based on the sensitivity (as represented by a label) of the information contained in the objects and the formal authorization (e.g. clearance) of subjects to access information of such sensitivity'.

The converse of mandatory access control is *discretionary access control*, which is defined as 'a means of restricting access to objects based on the identity of the subject and/or groups to which they belong. The controls are discretionary in the sense that a subject with a certain access permission is capable of passing that permission (perhaps indirectly) to any other subject'. The BLP model supports mandatory access control by determining access rights from the security levels associated with subjects and objects. It also supports discretionary access control by checking access rights from an access matrix. With respect to specification, we can regard the multi-level model as adding higher-level mechanisms to the matrix model. In addition to supporting arbitrary access specifications to the access matrix, the model groups protected objects according to different security labels and decides user privileges by their authorized security clearance levels (it is awkward, though not impossible, to specify this kind of access definition using the matrix model.).

In a practical data processing system, one approach that has been the subject of much research and development to enforce the BLP model is based on the *reference monitor concept* as illustrated in Figure 1. The reference monitor is a controlling element in the hardware and operating system of a computer that regulates the access of subjects to objects on the basis of their security parameters. The reference monitor has access to a file, known as the *security kernel database*, that lists the access privileges (security clearance) of each subject and the protection attributes (classification level) of each object. The reference monitor enforces the security rules (no read up, no write down) of the BLP model and has the following properties:

1. Complete mediation: The security rules are enforced on every access, not just, for example, when a file is opened.

2. Isolation: The reference monitor and databases must be protected from unauthorized modification.
3. Verifiability: The reference monitor's correctness must be provable. That is, it must be possible to demonstrate mathematically that the reference monitor enforces the security rules and provides complete mediation and isolation.

These are stiff requirements. The requirement for complete mediation means that all access to data within main memory and on disk and tape must be mediated. Pure software implementations impose too high a performance penalty to be practical. The solution must lie at least partly in hardware. The requirement for isolation means that it must not be possible for an attacker, no matter how clever, to change the logic of the reference monitor or the contents of the security kernel database. Finally, the requirement for mathematical proof is formidable for something as complex as a general-purpose computer. A system that can provide such verification is referred to as a trusted system. In Figure 1, there is also an audit file. Important security events, such as detected security violations and authorized changes to the security kernel database, are stored in the audit file.

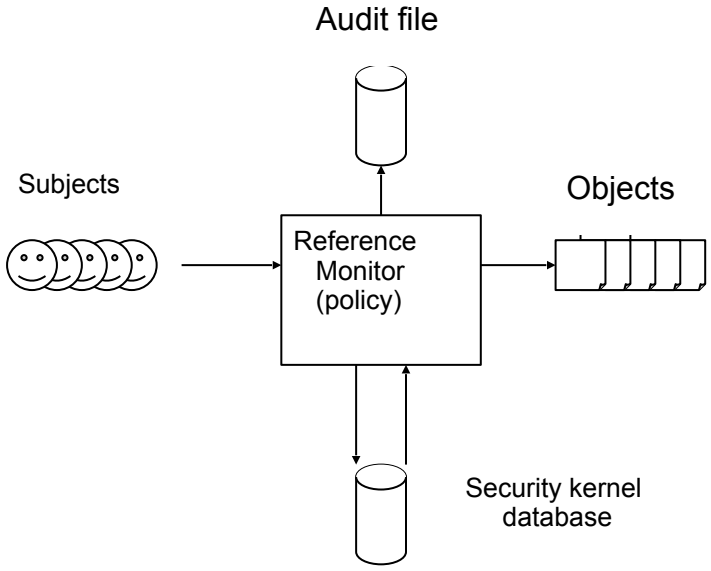

Figure 1. Reference Monitor Concept

In an effort to provide a service to the public, the US National Security Agency (NSA) established the Computer Security Center in 1981 with the goal of encouraging the widespread availability of trusted computer systems. This goal was realized through the Center's Commercial Product Evaluation Program (e.g. see the Orange Book). In particular, the Center attempts to evaluate commercially available products as meeting the security requirements that we discussed above. It also classifies evaluated products according to the range of security features that they provide. These evaluations are

published and freely available. Thus they serve as guidance for commercial customers for the purchase of products.

The Biba Model

From our crypto lecture, we learnt that confidentiality and integrity are in some sense dual concepts - i.e. confidentiality is a constraint on who can read a message while integrity is a constraint on who may have written or altered it. The BLP model is for the protection of confidentiality. In studying the two properties of the Bell-LaPadula model, Ken Biba proposed the Biba model for the protection of integrity. The Biba integrity model is the exact opposite of BLP and is often referred to as BLP upside-down. It requires that, for an integrity system, we must only read up and write down. But it should never read down or write up, as either could allow high-integrity objects to become contaminated by those of low-integrity.

The Clark-Wilson Model

Confidentiality, integrity, and availability are three essential properties for both military and commercial information security systems. In a military environment, the main objective is to prevent disclosure of information. For a commercial system (like a bank system), however, the main concern is to ensure that data integrity is protected from improper modifications and inappropriate actions performed by unauthorized users. The Clark-Wilson security policy model seeks to formalize the principles of accounting security that have accumulated over centuries of experiential bookkeeping. The Clark-Wilson (CW) model consists of subject/program/object triples and rules about data, application programs and triples. In the following, we will briefly discuss the triples and rules.

All formal access control models that pre-date the Clark-Wilson model use the concept of an ordered subject/object pair — that is, a user and an item or collection of data, with a fixed relationship (e.g. read or write) between the two. Clark and Wilson recognized that the relationship could be implemented by an arbitrary program. Accordingly, they devised an ordered subject/program/object triple. They use the term transformational procedure (**TP**) for a program to make it clear that it has integrity-relevance because it modifies or transforms data according to a rule or procedure. Data modified by transformational procedures are called constrained data items (**CDI**). This is because they are constrained in the sense that only transformational procedures may modify them and that integrity verification procedures (**IVP**) exercise constraints on them to ensure that they have certain properties, of which consistency and conformance to the real world are two of the most significant.

Unconstrained data items (**UDI**) are all other data - chiefly the keyed input to transformational procedures. Once subjects have been constrained so that they can gain access to objects only through specified transformational procedures, transformational procedures can be embedded with whatever logic is needed to effect limitation of privilege and separation of duties. Transformational procedures can themselves control access of subjects to objects at a finer level of granularity than that available to the system. What is more, they can exercise finer controls (e.g. reasonableness and consistency checks on unconstrained data items) for such purposes

as double-entry bookkeeping, thus making sure that whatever is subtracted from one account is added to another. To be specific, access control is by means of triples (subject, TP, CDI) which are so structured that a shared control policy is enforced. According to Amoroso's formulation (as illustrated in the textbook):

1. The system will have an IVP for validating the integrity of any CDI
2. The application of a TP to CDI must maintain its integrity
3. A CDI can only be changed by TP
4. Subjects can only initiate certain TPs on certain CDIs
5. Triples must enforce an appropriate separation of duty policy on subjects
6. Certain special TPs on UDI can produce CDIs as output
7. Each application of a TP must cause enough information to reconstruct it to be written to a special append-only CDI
8. The system must authenticate subjects attempting to initiate a TP
9. The system must only permit special subjects (i.e. security officers) to make any authorization-related lists.

We can split these principles into two categories: well-formed transactions and separation of duty.

Separation of duty states that no single person should perform a task from beginning to end, but that the task should be divided among two or more people to prevent fraud by one person acting alone.

A well-formed transaction states that, in a transaction, the user can only manipulate data in a constrained way. A security system in which transactions are well formed ensures that only legitimate actions can be executed. This ensures that internal data is accurate and consistent to what it represents in the real world.

Let's conclude our discussion of the CW model with a small example.

Alice creates an order for a supply, sends copies to the supplier and to the receiving department. Upon receiving the goods, Bob checks the delivery and signs a delivery form. Both the delivery form and original order go to the accounting department. The supplier then sends an invoice to the accounting department. Carol, who works in the accounting department, compares the invoice with the original order and delivery form and issues a check to the supplier. We can interpret the above steps using the CW model. The users are Alice, the supplier, the receiving clerk, Bob and Carol. The transformation procedures are 'create order', 'send order', 'create delivery form', 'send delivery form', 'sign delivery form', 'create invoice', 'send invoice', 'compare invoice to order', and so on. The constrained data items are order, delivery form, invoice and check. Users may only invoke some Transformation Procedures, and a pre-specified set of data objects or CDIs.

The Chinese Wall model

There are several security policy models that generalize that devised by Clark and Wilson. One of these, the *Chinese Wall* security policy model, is perhaps as significant for some areas of the commercial world as Bell and LaPadula's model is to the military. Frequently there is a need to manage conflicts of interest. An example from the textbook is as follows: an advertising copywriter who has worked on, say, the Shell account, will not be allowed to

work on any other oil company's account for some fixed period of time. Unlike the Bell and LaPadula model, access to data in the Chinese Wall model is not constrained by attributes of the data in question but by what data the subject already holds access rights to. Essentially, datasets are grouped into 'conflict of interest classes' and by mandatory ruling all subjects are allowed access to at most one dataset belonging to each such conflict of interest class. The actual choice of dataset is totally unrestrained - provided that this mandatory rule is satisfied. It seems hard to model such policies in the BLP model.

In order to further understand the practical issues of those security policy models we have explored here, you may wish to refer to the details of SWIFT and ATM protocols as outlined in the textbook.

Monitoring Systems

In this section, we briefly discuss some practical examples. (Although it is not mandatory to read the text, if you are sufficiently interested, read the grayed boxes "HOW TO STEAL A PAINTING (1-7)" in Section 10.2). It should be noted that these example protocols are not really related to computer systems. However, the principles are the same. From analysis of these protocols, we can learn that threat models play an important role in the process of building a secure system. We will also learn about denial of service attacks in the real world.

The textbook analyzes several types of burglar alarm systems and presents ways of breaking these protocols. In summary, these protocols fail for the following reasons:

1. There is a backdoor for the thief. For example, if the alarm systems are only installed at floor level, then the thief could come in and out via the roof. If there is an emergency exit close to the armed painting, then the thief could easily steal the painting without being caught by the police since by the time the police arrive, the thief will already have escaped to safety.
2. The sensor is defeated. This is always possible if the thief knows how to prevent the sensor from working. Unfortunately, it is feasible to defeat most sensors.
3. A denial of service attack is always successful in practice. The textbook lists a number of examples. For example, make the alarm falsely alarmed several times before the thief really steals the paintings (by this time, police thinks that the alarm is falsely alarmed again), or attack the communications between the sensors and alarm controllers.

The lessons we have learned regarding burglar alarm system failure are also important for computer systems. Indeed, most reported attacks fall into the following three categories: back door, system defeat, and denial of service.

Nuclear Command and Control

In this section, we briefly discuss the three essential components of nuclear command and control: unconditionally secure authentication code, shared control schemes, and prescribed action links. We discussed subliminal channels in Week 1. Thus we will not talk

about treaty verification and subliminal channels in this lecture. You are nevertheless recommended to read the corresponding section in the textbook.

Authentication code

In a military system, authentication should be secure unconditionally. Note that we studied digital signatures and message authentication code in the crypto week. However, these schemes are generally conditionally secure. That is, it is only secure if the underlying assumption holds. For example, an RSA signature is secure against forgery attacks only if factorization is hard (which is only an assumption). Imagine then that a country uses the RSA digital signature authentication scheme to issue military commands and some other country has a secret method of forging RSA signatures. The result, obviously, would be disastrous. Thus for military usage, we need an unconditionally secure authentication method. In another words, we need an authentication method that is secure in the sense of information theory. A simple unconditionally secure authentication scheme can be constructed as follows. (Note that unconditionally secure authentication schemes are always symmetric).

Assume that Alice and Bob share two elements a and b from a finite field, e.g. from the set {1, 2, …, p-1} where p is a large prime. In order for Alice to send an authenticated message m to Bob (we assume that m is also an element from the same finite field), Alice could send the pair (m, am+b) to Bob. After receiving a message pair (m,c) from Alice, Bob checks whether c=am+b. If the equation holds, then Bob is assured that the message came from Alice, otherwise, Bob regards m as a forged message. A simple mathematical calculation shows that Carol, who knows nothing about the secret key (a,b), cannot forge any authenticated message no matter how much computation power she has. We should also note that one key pair (a,b) could only be used to authenticate one message, since if it is used twice, then Carol, who saw the two messages and their authenticators, could recover the secret key (a,b). You should also be aware that this kind of authentication method does not provide all properties such as those provided by the public key authentication method (e.g. digital signatures). However, for many military command systems, this authentication method is good enough.

Shared control schemes

Secret sharing schemes are widely used both in military and civilian systems. For example, control of nuclear weapons in Russia involves a two-out-of-three access mechanism. The three parties involved are the President, the Defense Minister, and the Defense Ministry. Threshold schemes like this one involve a special type of secret sharing. Formally, let $t < n$ be positive integers, a t-out-of-n-threshold scheme is a method of sharing a key K among a set of n participants in such a way that any t participants can compute the value of K, but no group of t-1 participants can do so. The textbook provides a nice description of a shared control scheme using geometry. Another commonly used shared control scheme is the Shamir secret sharing scheme that we describe here. Let P(x) be a random polynomial of degree t-1 in which the constant term is the secret key K. Every participant P_i obtains a point (x_i, y_i) on this polynomial (that is, $y_i=P(x_i)$). Let us look at how a subset B of t participants can reconstruct the key. This is basically accomplished by means of polynomial interpolation.

Suppose that participant P_1, P_2, P_3, …, and P_t want to determine K, they know that

$$y_i=P(x_i)$$

for all $1 \le i \le t$. Since P(x) has degree of t-1, P(x) can be written as

$$P(x)=K+a_1x+\ldots+ a_{t-1}x^{t-1}$$

where the coefficients K, a_1, …, a_{t-1} are unknown elements. Since $y_i=P(x_i)$, we can obtain t linear equations in the t unknowns a_1, …, a_{t-1}, where all arithmetic is done in the default field. If the equations are linearly independent, there will be a unique solution, and K will be revealed from these unknowns. Due to the Vandermonde property, these equations are linearly independent if these x_i are chosen to be different.

Example. Suppose that t = 3, n = 5, and all the mathematical operations are carried out mod p = 17. Now assume that P_1, P_3, and P_5 want to recover the secret key K and pool their shares, which are (1,8), (3,10), and (5,11) respectively. Writing the polynomial P(x) as

$$P(x)=K+a_1x+ a_2x^2$$

And computing P(1), P(3), and P(5), the following three linear equations in Z_{17} are obtained:

$$K+ a_1 + a_2 = 8$$
$$K+3a_1+ 9a_2=10$$
$$K+5a_1+ 8a_2=11$$

This system does have a unique solution in Z_{17}: K = 13, a_1=10, a_2=2. The key is therefore K = 13.

A more general situation is to specify exactly which subsets of participants should be able to determine the key and which should not. We will not go into details here of these generalized access structures.

Permissive action links (PAL)

A 'Permissive Action Link' is the box which is supposed to prevent unauthorized use of a nuclear weapon. For a detailed description of the history of PAL, types of PAL, cryptography and PAL, PAL and key management, and why PAL is classified, please see Bellovin's paper (the link is provided at the end of this lecture).

Some further links and references

1. The Orange Book. https://csrc.nist.gov/csrc/media/publications/conference-paper/1998/10/08/proceedings-of-the-21st-nissc-1998/documents/early-cs-papers/dod85.pdf
2. Official version of the BLP Model: https://www.geeksforgeeks.org/introduction-to-classic-security-models/
3. Edward Amoroso. Fundamentals of Computer Security Technology. Prentice Hall, 1994.

4. David D. Clark and David R. Wilson ’A Comparison of Commercial and Military Computer Security Policies.’ IEEE Symposium of Security and Privacy, 1987, pages 184-194.
5. S. Bellovin. Permissive Action Links. https://web.stanford.edu/class/ee380/Abstracts/060315-slides-bellovin.pdf
6. Purple Penelope, DERA: http://www.opengroup.org/security/meetings/sep97/Group.pdf
7. Purple Penelope: Extending the Security of Windows NT - http://www.opengroup.org/security/meetings/sep97/pp/ppweb4.pdf
8. Trusted Computing FAQ by Ross Anderson: http://www.cl.cam.ac.uk/~rja14/tcpa-faq.html