Name: Jessica Lucci
EID: jml3624
CS Login: jml3624
Email: jessicalucci14@gmail.com

Lecture 1

1. Personal security, computer security, school security, and home security are just a few of the uses of the term security that are relevant to my everyday life.
2. Each of these uses involve the protection of either myself or my assets against some type of threat (whether it be real or plausible).
3. I have been the victim of lax security on a few occasions, such as when I forgot to lock my car once and someone stole some of my electronics in the car while it was parked and unattended.
4. The likelihood of my laptop being infected seems to be relatively low. I have installed 3rd party software from the web before, but only if it had a large number of reviews from users, or a few news articles written in regards to it. Additionally, my laptop appears to function completely normally. I don't get strange pop-ups, random programs closing, etc, as I have when my computer has been infected in the past.
5. I don't have any malware detection software installed or anything like that, so the preventative measures I take are just to not install/download anything that I cannot verify in some way.
6. They have seemed to be effective so far. I have never had a virus (known to me) on this laptop before.
7. I don't believe the FBI agent's quote is an overstatement. As the sophistication of technology increases (along with its' widespread use), more and more data is being stored on computers, and more and more networking systems are being created. This means that more connections are being created between systems, and more data is coming online. This creates a continuously greater possibility that someone with a malicious intent could access potentially life-altering information
8. By learning about computer security we can better protect ourselves from computer-related threats.

Lecture 2

1. Another reason security is hard is that many systems often rely on the security of other systems to perform correctly and efficiently, and oftentimes don't know how those other security systems function. For instance, if I'm setting up an online store I may use a tool like paypal to manage my secure monetary transactions. The security of my site depends entirely on the security of paypal - if they're compromised, my system is now open to threats. Additionally, since I haven't looked at paypal's transaction code before, I might accidentally write something

into my code that is completely valid from paypal's point of view, but allows for malicious use of my site.

2. There is no way to systematically enumerate the bad things that may happen to a program, as there is no possible way to know what all of those bad things are. You might think you've accounted for every possible attack, and your code might be working perfectly, but there's always going to be a chance you've overlooked some sort of security compromise
3. Attackers only have to find and exploit one weakness in a system, whereas a defender has to find and eliminate all exploitable weaknesses. It only takes one thing to compromise a system, so a defender has to constantly be on the lookout for whatever that one thing may be.
4. I agree with the quotes from Morris and Chang as it's just physically impossible to know every single threat that may ever happen to you. Even when you've triple checked your code, encrypted every piece of data five times over, etc, there's *always* going to be a possibility that you've missed something.
5. Since it's impossible to build a perfectly secure system, there has to be a tradeoff between the effort put into securing a system and the effort put into making other parts of the system work, improvements, etc. Otherwise, you'll spend your entire time trying to make the system secure, and nothing else would ever happen with the system.

Lecture 3

1. Risk is "the possibility that a particular threat will adversely impact an information system by exploiting a particular vulnerability"
2. I believe that "in the real world" software security is about risk management (since we cannot build a perfect secure system), but in the more technical sense, I believe that software security is about risk management *without* having to prioritize risk and make security-related decisions based on that prioritization. Software security should, ideally, protect against *all* threats equally.
3. I accept the risk of getting a speeding ticket when I drive over the speed limit, I avoid the risk of having an allergic attack by staying away from cats, I mitigate the risk of having my house broken into by locking my doors when I'm not home, and I transfer the risk of having to pay large hospital bills by paying for health insurance.
4. ALE functions as a risk management tool as it allows for prioritization of risks in regards to money. Risk prioritization can rely on many other things than monetary loss alone, but the ALE model allows for the monetary factor to be appropriately accounted for during any sort of prioritization.
5. Factors such as money, health/life and value of assets are all rational things to consider during risk assessment.

Lecture 4

1. The key distinction between the things listed on slides 2 and 3, is that the things listed on slide 2 are *goals*, and things listed on slide 3 are *mechanisms*. In regards to computer security, we are

trying to solve these goals, not the mechanisms. The mechanisms are only tools for protecting for the goals.

2. In my personal life, protection of confidentiality is most important. I'm mostly concerned with malicious parties being able to access information like my credit cards numbers, or social security number.
3. Data is often grouped by level of security (how sensitive is the data) and categorized within that group. For example, data may be grouped by "top secret" and "secret" sensitivity levels, and then categorized into "army", "navy" and "marines" within one or more of those groups. So instead of having all of the data in one area, splitting it up makes it easier to protect and access.
4. Authorizations might change over time as people get promoted/fired, data no longer becomes relevant, etc.
5. Availability means that *what* I need available is available *when* I need it - a system must reliably supply the needed resource *when* it's needed.
6. Authentication is important when you want to make sure the person trying to access some asset is actually the person they say they are. For instance, you don't want anyone that's not you to be able to log onto a website pretending to be you that contains your sensitive information (like credit card numbers). Non-repudiation is important when you want a guarantee that a transaction that took place will not be erased. For instance, if you buy something off a website, you want a guarantee that the website won't tell you that transaction never happened after you've paid them for your item.

Lecture 5

1. Cell Phone Network Metapolicy: User's cell phone numbers should be protected from disclosure; Military Database Metapolicy: Government secrets should be protected from disclosure to unauthorized parties
2. A metapolicy is just an abstract overview of the security goal - it doesn't provide any concrete goals, and is oftentimes too general to be of use. The policy provides the specific and enforceable guidelines to support the metapolicy.
3. Three possible rules regarding student records:
    a. Once a grade has been entered into the system by an administrator, it may not be changed
    b. Student records may only be released to the student, or a party that the student authorizes
    c. Student records may not be shared with anyone without the consent of the student
4. Stakeholder's interests oftentimes differ in a policy. For example, in regards to protecting student records, imagine a policy existed that stated "once a grade has been entered into the system by an administrator, it may not be changed". The student, one stakeholder, supports this policy because they don't want their grade to be potentially lowered. The school, another

stakeholder, supports this policy because they don't want the change of grades to reflect poorly on their institution.

5. The likely metapolicy is probably something along the lines of "student's social security numbers should be protected from disclosure".
6. If you don't understand a metapolicy, you can't establish or describe any sort of policies to help protect that metapolicy. Additionally, even if you did write a policy, you wouldn't be able to justify its' enforcement - you can't explain why it was written in the first place!

Lecture 6

1. Military security is mostly about confidentiality because the protection of military information from malicious parties is one of the main ways the military serves its' function of protecting its' nation. There are, however, aspects of integrity (you don't want unauthorized parties to be able to change the date or place of attacks, etc) and availability (you want to be able to access things like location of troops at appropriate times).
2. The major threat in the MLS thought experiment was someone seeing a document for which they were not authorized to see.
3. This proviso is included because some security measures that are put in place that satisfy confidentiality sometimes do not satisfy (or even negate) integrity or availability.
4. The labels we use have an element taken from a linearly ordered set of security levels, and an element taken from an unordered set of "need to know" categories.
5. Since we're only concerned with confidentiality, we don't care how or if the information is labeled correctly - only that the labeled data is not seen by anyone that does not have the proper credentials to do so.
6. Facts listed by decreasing sensitivity:
    a. The Normandy invasion is scheduled for June 6
    b. The British have broken the German Enigma Codes
    c. Col. Jones just got a raise
    d. Col. Smith didn't get a raise
    e. The cafeteria is serving chopped beef on toast today
    f. the base softball team has a game tomorrow at 3pm
7. Labels for facts (in same order as #6)
    a. Top Secret: {Invasions, Foreign Affairs}
    b. Top Secret: {Foreign Affairs}
    c. Confidential: {Salary}
    d. Confidential: {Salary}
    e. Unclassified: {Cafeteria}
    f. Unclassified: {Recreational Sports}
8. If a document contains two different levels of sensitive information, you always use the higher level of sensitivity for labeling. This prevents people without security clearances high enough to

view that higher level information from accessing the document. If there are multiple categories associated with the document, all the categories must be used in labeling to prevent people without clearance for viewing any of the categories from accessing the document.

Lecture 7

1. Labels are affixed to humans through their clearance or authorization level.
2. Labels on documents indicate the sensitivity of the enclosed documents, whereas labels on humans indicate the types/levels of information that human is allowed to access.
3. In the context of computers, documents may be seen as data or files, and humans may be seen as user accounts, or a set of credentials.
4. The principle of least privilege prevents unnecessary security risks from being created.
5. Access Table
    a. The human has a Secret clearance, which is higher than Confidential, and the human is cleared to access all things Crypto. Therefore they should be able to see Confidential: {Crypto}.
    b. Even though the human has clearance to the Crypto category, their clearance is only Secret, so they should not be able to view a document labeled Top Secret: {Crypto}.
    c. Since the human has a Secret clearance, they should be able to see all documents labeled Unclassified: {}. Even though they have clearance to the category Nuclear, {} represents all un-categorized documents, so their Secret clearance accounts for their access to Unclassified: {}.

Lecture 8

1. The vocabulary terms objects, subjects and actions were likely introduced to provide a more generalized view of security policies, as so far we've been using concrete examples of humans and documents.
2. Partial Order of Dominates Relation Proof
    a. Reflexive
        i. Given some label X, and some label Y, such that X = (L_1, S_1), and Y = (L_1, S_1), X = Y
    b. Transitive
        i. Given some label X, some label Y, and some label Z, such that $X \geq Y$ and $Y \geq Z$, we know $L_x \geq L_y$ and $S_y \subseteq S_x$ and $L_y \geq L_z$ and $S_z \subseteq S_y$ from the dominates definition. So, we have $L_x \geq L_y \geq L_z$ and $S_z \subseteq S_y \subseteq S_x$. Then, since $L_x \geq L_z$ and $S_z \subseteq S_x$, $X \geq Z$ by the dominates definition.
    c. Anti-Symmetric
        i. Given some label X and some label Y such that $X \geq Y$ and $Y \geq X$, we have $L_x \geq L_y$, $S_y \subseteq S_x$ and $L_y \geq L_x$, $S_x \subseteq S_y$ by the dominates definition. Then, ($L_x \geq$

$L_y) \wedge (L_y \geq L_x) \rightarrow L_x = L_y$ and $(S_y \subseteq S_x) \wedge (S_x \subseteq S_y) \rightarrow S_x = S_y$. Since $L_x = L_y$ and $S_x = S_y$, $X = Y$

3. Given label $X = (L_1, \{C_1\})$ and label $Y = (L_1, \{C_2\})$, X does not dominate Y because $S_y = \{C_2\}$ is not a subset of $S_x = \{C_1\}$, and Y does not dominate X because $S_x = \{C_1\}$ is not a subset of $S_y = \{C_2\}$.
4. The simple security policy says that a person may read a document if their label dominates the document's label.
5. The simple security policy uses "if" instead of "if and only if" as label dominance may not always be the only thing preventing/granting someone access (for example, you might have a high level of clearance, but you may also currently be on a "do not give access" list for malicious behavior, etc).

Lecture 9

1. Simple Security only deals with read access. Someone could copy data under label X into a document under label Y, where X dominates Y. So even though that data should only be read by people with access to label X, it can now be read by people with access to label Y as well, violating confidentiality.
2. We need constraints on write access to make sure people don't write to labels they're not also cleared to read from.
3. Write access is especially important in the realm of computers, as "people" are represented by programs operating on behalf of a user. I might authorize and run a program that I think is safe, but in actuality has been compromised by someone with malicious intent that has changed that program to write out sensitive information to an unauthorized party.
4. The *-Property says that a subject may only write to an object if that object's label dominates the subject's label.
5. For a subject to have both read/write privileges on an object, the subject's label must match (be equal to) the object's label.
6. To deal with this issue, the general would need to log out of his top secret account, log back into an unclassified account, and then send the orders to the the other unclassified users from that account.
7. A corporal overwriting a war plan is a problem of integrity, not confidentiality. So the *-Property holds for confidentiality. If we wanted to prevent this behavior, we'd need to make a rule in regards to integrity, such as putting specific write restrictions on documents (Ex. Only users X,Y,Z can edit this document, or only users with security clearance above X can edit this document, etc.).

Lecture 10

1. Changing a subject's level up is probably "bad" as the subject now has read access to documents he did not have before. Changing a subject's level down can be "bad" if the subject carries high level information down with them, as this is the equivalent to a write down.
2. If subjects and objects could not change labels during the lifetime of the system (strong tranquility), every time you needed to move either an object or subject (such as a private being promoted to a general, or wanting to release the results of a previously top-secret drug experiment to the public), you would have to delete the current subject/object and then create an entirely new subject or object with a new label. This takes up a lot of unnecessary time, and could potentially compromise things like availability if you have a user that has to operate at different levels in a short period of time (system may not be able to keep up with delete/recreate).
3. Lowering the level of an object might be dangerous because you allow access to the document be previously unauthorized users - it's equivalent to a read up.
4. An object downgrade must not carry any residual higher-level information with it for a downgrade to be secure.

Lecture 11

1. If you wanted to build a system in which all subjects had read but no write access to all files, you could give all users a security level X, and all files security level Y, such that X > Y.
2. You usually wouldn't build an access control matrix for a BLP system, as the matrix would be much too large for most realistic systems.

Lecture 12

1.
    a. (H, {}) → (H, {A})
    b. ↑ ↑
    c. (L, {}) → (L, {A})
2. For any two labels in a BLP system, their LUB and GLB may be found by following the direction of the arrow branching off of the labels.
3. If our lattice has anything but an upward flow, we know there's been a violation of some security goal. Since the metapolicy for any BLP system is to constrain the flow of information among the different security levels, the upward flow of our lattice represents just that.

Lecture 13

1. BLP rules control the flow of information in a system. So in the metapolicy of the first example, the BLP rules prevent information from flowing from H to L, since H > L.
2. READ(S,O) satisfies the BLP properties as it does not allow information to flow down the system (from higher to lower security levels). WRITE(S,O) also satisfies the BLP properties as

it does not allow information to flow down in the system either. Subjects can only send information up (or at their level).

3. CREATE(S, O) satisfies BLP properties as it is equivalent to writing at the same level of the subject, which is allowed by the *-star property. DESTROY(S,O) also satisfies the BLP properties as it is equivalent to writing up, in which the "writing" is actually just removing all the content from a file.
4. In order for the covert channel to work, there must be some sort of synchronization mechanism between the two subjects - S_L has to know when to try and read the modified file.
5. The DESTROY command is at the end of the statement in order to prepare for the next bit transmission. If the file wasn't destroyed all future CREATE commands would fail since the file would already be in existence, and the covert channel wouldn't work.
6. In the first path, the file is empty, and in the second path, the file has a 1 written in it.
7. S_L does the same thing in both cases so that the content of the file depends solely on S_H's actions. In this particular case S_L must do the same thing in this scenario since S_H is changing the variables. If S_L did different things they wouldn't know if S_H was trying to send a 0 or 1 since they wouldn't know if they overwrote some S_H's change.
8. S_H does different things so that it can send different bits to S_L. In this case S_H must do different things. If they did the same thing S_L would get the same bit every time, and no message would be sent.
9. If S_L sees a varying result depending on an action taken by S_H that is used to send a bit of information from S_H to S_L, the metapolicy has been violated as information has flowed down.

Lecture 14

1. Talking over coffee is not a covert channel because a covert channel must be a flow between subjects in a system. The people talking over coffee are not part of the system.
2. That is not a covert channel as the information flow does not occur via a system resource (file attributes, flags, etc) that were *not* intended as communication channels. The information is transferred via a system resource that *was* intended as a communication channel - the file contents.
3. The bit of information resides in the *Resource Not Found* and *Access Denied* error messages.
4. The bit of information resides in the time elapsed in the system clock.
5. The bit of information resides in the cylinder read order.
6. The bit of information resides in the value of l.
7. Termination channels may have low bandwidth as they rely on the termination of a computation. Therefore much of the bandwidth of the channel is taken up by the running computation itself.
8. In order to implement a power channel a higher level subject would have to be able to modulate the amount of energy consumed or produced by a power source, and a lower level subject would have to have some way to monitor that power source.

9. Power channels could arise for any computational hardware system. A higher level subject could run a program on some piece of hardware that requires different amounts of computational power at different times and a lower level subject could simply monitor the energy consumption of that piece of hardware.

Lecture 15

1. Covert channels on real processors can operate at thousands of bits per second with no notable impact on system processing. This is more than enough time to transfer large amounts of data.
2. It is infeasible to eliminate every potential covert channel as some channels that are abused are a necessity to the system.
3. Once a potential covert channel is identified it can be eliminated by modifying the system implementation, reduce the channel bandwidth by introducing large amounts of noise into the system or it can be monitored for patterns of usage that indicate someone is trying to exploit that channel.
4. A covert storage channel could exist if a file exists somewhere in the system that a higher level subject has access to modify, and a lower level subject has access to read. Additionally the two subjects would need some sort of mechanism for communicating with one another so they know when to modify/read the file in question.
5. This covert channel can be utilized by the sender and receiver to send a series of bits to one another in the form of a changeable attribute of the file, such as the size of the file, existence of the file, etc.

Lecture 16

1. An R would mean that the file existence operation could reference, or provide information about, the create attribute. Since the create operation doesn't return any information saying whether or not the file existed in the first place, it doesn't provide us any information about file existence. We only know that the file exists after we run create.
2. An R and an M in the same row in the table indicate that a covert channel may exist because the ability to both modify and reference that attribute exists. This is the framework needed to create a covert storage channel.
3. If an R and an M are in the same column a potential covert channel is not indicated. The R and M must be associated with a single attribute, and an R and M in a column would be associated with two different attributes.
4. Creating an SRMM table would allow someone to determine where covert channels may exist. For an administrator of the system this would show where potential threats may live so they would know where to investigate or protect against malicious users. For a malicious user of the system, a SRMM table would give them an idea of where to set up a covert channel.