Sploit 3: Multiple Sploit Calls

This exploit takes advantage of two vulnerabilities and has two steps to getting root access. The first part of the sploit uses the fact that if a hard link from the source file to the destination file in `copy_file` can be established, then the source file will be unlinked.

The vulnerability here is that any file can be removed from its current location and put into the submit directory. By having the sploit move to the `/etc/pam.d` directory, the first call to submit can be passed the `su` file, removing it from this directory.

Linux-PAM (Pluggable Authentication Model) is a system of libraries used to manage authentication of services on a Linux system such as the virtual machine. The su configuration under PAM specifies which groups or users for example, can use su and even use su without a password. Currently the only one who can use passwordless sudo is root, defined by the line:

```
auth sufficient pam_rootok.so
```

which tests if the user using su is root, and if they are, then that is sufficient to let the su command execute without a password.

The goal in this exploit is to somehow get the following line in the file as well:

```
auth sufficient pam_permit.so
```

which will make it sufficient for any user to execute su without a password and get to the root user.

Now that the old `/etc/pam.d/su` file has been removed, the second part of the sploit creats a new one.

The second part of the sploit looks at the vulnerability introduced by the `log_message` function. As long as the `submit.log` file does not exist in the user's home directory, then the `log_message` file will open the file name and write the message to it.

By making a symlink from `submit.log` to `/etc/pam.d/su`, the logging of the message will create a user-owned file that can contain any su configuration.

After running submit for a second time with any dummy submission and message, the file will be created and can be modified by the rest of the sploit to include the new PAM configuration.

As soon as the new configuration is saved, the user can then `su root` and gain password-less access to the root user.

This exploit uses two powerful vulnerabilities exposed in the submit application, the first being that any file in the system can be removed from its current location and put in the submit directory, and second that new files can be created anywhere in the system.

Each of these vulnerabilities have restrictions:

In the first one, the file that is submitted to the submit directory and removed from where it currently is cannot contain /bin/sh, since that is checked by the `check_for_viruses` function. If this wasn't the case, then a great target for the exploit would have been /etc/passwd.

The second vulnerability is also restricted because the submit program does not let you modify privileged files, since that is caught by the `get_logfile_name` function. In this exploit, a way to remove a file not mentioning /bin/sh and which could later be replaced had to be found.

This exploit introduces many other possible ways to gain root access, as all that is required is finding a file that does not mention /bin/sh and is important for user permissions.

To fix the first vulnerability in the exploit, the submit program could make sure that the files that can be submitted are only the ones allowed to be accepted as assignment files, similar to how the current UW submit program works. This change could be made in the `copy_file` method. It should also use `realpath` to make sure that the file linked isn't a link itself to an important system file.

The second vulnerability can be fixed by first checking the `realpath` of the `submit.log` file in `log_message` to make sure it isn't a link to somewhere else either.

Sploit 4: Directory Path Exploit

This exploit focuses on the vulnerability of the submit program created by `get_submit_dir()` function, which creates a new submit directory for the user submitting the file if no directory already exists.

The vulnerability is that the program uses `run_cmd("mkdir", ...)` instead of `run_cmd("/bin/mkdir", ...)`. This vulnerability allows an attacker to modify the environment of the program being run to prepend `.` in the PATH environment variable, meaning that the `mkdir` command that the submit program intends to use can be replaced with any other executable named `mkdir` in the current directory, which will then take precedence.

This is accomplished by first modifying the path, and then making an executable file called mkdir which on execution runs a shell that will provide root access to the user.

This vulnerability can be fixed by ensuring that everywhere `run_cmd` is used, it references the program to run absolutely, rather than relatively, so that it will always use the intended version of the specified program.

Written Questions

1.

a)

The installation of spying software by one's partner is a violation of one's confidentiality.

Confidentiality pertains to who is authorized to access certain data. If a person installs spying software on one's partner, then it is likely the case that they are trying to gain authorization to information that one's partner has chosen to keep confidential. This attempt at gaining unauthorized access to one's partner's data therefore compromises the partner's confidentiality.

This is a security issue, not a privacy issue. A privacy issue is related to the individual's right to informational self-determination and choosing who gets to use, see and give away his or her data. The correct privacy policies were in place, the person was not permitted to access the data, but the security of the phone was compromised, allowing access to data that should have remained private. It is also not an availability or integrity compromise, as all of the phone's data is still accessible to the partner and has not been altered.

b)

Google selling demographic information to malicious hackers is a violation of privacy.

This issue is a privacy violation, as Google's privacy policy to its users ensures that the data will not be given to malicious parties. To sell the data to malicious hackers is a violation of the privacy rights between Google and its users.

This is not a confidentiality issue as security of the Google users' data has not been breached, rather it was willfully given to the malicious hackers. This issue is not an availability compromise since the data is still available, nor is it an integrity compromise, since user data has not been modified and is still accurate.

c)

Breaking into a dating site and stealing user information is a compromise of confidentiality.

This is a security issue, and compromises confidentiality since the attackers have gained unauthorized access to data that they are not allowed to see.

This is not a privacy issue, since the privacy agreements between the site and the users have not been violated, rather a security breach that accessed data that was explicitly private to certain users was made. This is not an availability or integrity compromise, since the data is still present in the system and unchanged.

d)

Replacing the software in one's car is an integrity and availability compromise.

This is an integrity compromise, as the user expects certain software and firmware to be present in the car, possibly with features that they regularly used and which may no longer be present. Since the data in the car has been modified, this is an integrity compromise of the software and firmware of the car.

This is also an availability compromise, since certain features and data that was accessible in the car before the hack, such as voice navigation, or automatic parallel parking may now be broken as a result of the software and firmware replacement. The availability of abilities that the previous software and firmware supported has been compromised.

This issue is not a privacy or confidentiality compromise, since no user data has been noted as stolen or viewed by the hackers.

e)

The replacement of the expected site with this illegal one is a compromise of users' integrity, and availability.

This is an integrity compromise because now when the users of the site perform certain operations, they don't get the data they expect back, they get the malware.

Furthermore, this is an availability compromise as the website that the user wanted to use is no longer available and the purpose they had for using the original site is now not achievable. The data they had on the original site could also not be available after the site was replaced with the illegal one.

This does not compromise confidentiality or privacy since they were illegally using the site and the government is authorized to any data and information regarding the users in regard to their illegal activities.

f)

Lizards shutting down the Internet is an availability compromise, and frankly very scary. It is an availability compromise since the data on the Internet that users intend to access can no-longer be reached.

This is not any of the other compromises, as the Internet has only been made inaccessible and no data has been accessed or used by the lizards thus far.

2.

a)

The first threat that applies to the distribution of medication to patients is that the primary source for a component of the medication has had a malfunction and it is not able to continue production for a period of time. This threat is categorized as an interruption threat.

A second threat is that the medication is laced with unwanted or dangerous medications either by accidental contamination or intentionally in order to reduce

costs by using cheaper substitutes. This threat is categorized as a modification threat.

A third threat is that the medication is completely fabricated by an illegitimate source and sold to distributors, posing as the original medication. This threat is categorized as a fabrication threat.

A fourth threat is that during the distribution process, the medication is stolen and sold on the black market to patients or non-patients who may be addicted to the substance or require it for similar reasons. This threat is categorized as an interception threat.

b)

Prevent:

One way to prevent the threat of interruption of distribution to patients is to ensure that all steps in the process of getting the medication from production to distribution and ultimately to the consumer have redundancies. If medication cannot be distributed to certain patients, for example insulin to people who require it because of diabetes, then the consequences are very severe. The drug must have multiple sources of production and many different distribution centers so that if something happens to one of them, there is always a way to get the drug to the people who need it.

Deter:

One way to deter the threats of interception, modification, and fabrication of the drug is to pose stricter laws surrounding these illegal activities. Many potential thieves and illegitimate modifiers or fabricators will re-consider their activities if the laws that they are breaking are made too severe to be worth the continued risk. This of course won't solve the problem completely, but it acts as a deterrent to some groups of people.

Deflect:

One way to deter the threat of fabrication is to release the drug for free. The fabricators will have no incentive to fabricate the drug since if they charge money for it, their fabrication will not be purchased over the free option already provided.

Detect:

A way to detect the threats of interception, modification and fabrication of the drug is to more tightly monitor the transport, creation and distribution of the drug along all steps of the production process. This can be achieved through cameras in all transport vehicles and production facilities, as well as marking of the drug with serial numbers that can be used to trace its origins.

Recover:

One way to recover from the threat of interruption is to boost production of the drug once whatever interruption that occurred has been resolved. If there

was a shortage of a certain chemical that stopped production for a day, then contact other centers to ship the ingredient as soon as possible. While waiting for the production to resume, increase production of all other drugs so that when production is ready to continue, resources can be diverted from the production of the other drugs to the one that experienced the shortage until the supply is restored.

3.

Duqu:

Duqu is a trojan with code very similar to Stuxnet. While Stuxnet was an aggressive program that altered the integrity of the computers it infected, Duqu is passive, and acts as a data-collector and back door for future attacks by giving attackers access to a machine.

Once Duqu is installed, it could contact its server and if instructed, perform keyboard-logging, network-sniffing, screenshotting, and other espionage software.

How it Spreads

Duqu takes advantage of a vulnerability in Windows TrueType font-parsing engine. The installer file for the trojan is a Microsoft Word document which exploits the vulnerability to gain root access to the infected machine.

ZeuS:

ZeuS is another trojan that targets Windows machines for the purpose of stealing banking passwords and financial information by keyboard-logging and network monitoring. It also creates a botnet, a network of corrupted machines that become under the attacker's control.

How it Spreads

It is installed by downloading shareware or through phishing emails that include malicious links and attachments.

Conficker:

Conficker is a worm that has infected over 9 million machines.

How it Spreads

A vulnerability in the Microsoft Server Service discovered in 2008 allowed attackers to send a specific RPC, remote procedure call, that triggers a buffer overflow and gives the attacker root access. When the packet is sent to port 139 or 445 on a Windows share, the machine becomes infected.

Once infected, an HTTP server is downloaded, which then scans for other target machines and sends the exploit to them as well. The malicious code is downloaded to infected machines from already infected machines, meaning that there is no one source of the attack.

Cryptowall:

Cryptowall is a trojan and ransomware that encrypts a computer's data using RSA-2048 and demands a ransom in order to receive the decryption key. Infected machines give the user 7 days to pay for the decryption, otherwise the ransom doubles.

How it Spreads

Cryptowall infects computers by being downloaded as an email attachment, through clicking ad links or as fake updates to applications.