COMP 4108 – COMPUTER SYSTEMS SECURITY

Assignment 2

Student Name: Ben Cendana
Student #: 100811212
Date: March 5 2018

Table Of Contents

Preface

References to each question are available in the Bibliography page.

1.0: What Is A Certificate

One of the biggest problems with internet security is authentication, authentication from a security perspective means that whenever a website is being viewed we need to verify if the source of that website is coming from a trusted source. If we know the source of the website is coming from a trusted source we can then form some level of trust. The principal behind building a level of trust forms the foundation of modern digital certificates.

As an example every student is provided with a student card and number, if a student is studying overnight in the library during the 24hr library exam period.

Staff may ask the student to authenticate themselves by presenting there student card. If a student provides a valid piece of identification staff will allow them to enter the library as they can form some level of trust with the student since they have been authenticated as a valid Carleton student. If a student fails to produce a valid student card they have no way of authenticating if the individual is actually a student or a stranger trying to access the library.

In essence a digital certificate is the electronic version of a certificate works in the very same way. The certificate provides some information about the website, its use, the owner and who is the authenticating authority for that website.

1.1: Digital Certificates

When websites provide a digital certificate we are provided with who is the authorized authority for that webpage, the source of that webpage and the encrypting algorithms being used. With a digital certificate we can then authenticate the source and form some level of trust by verifying the authorized authority and then authenticating them.

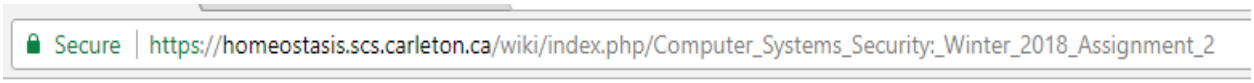

Figure 1: The Course Webpage With Secure Lock

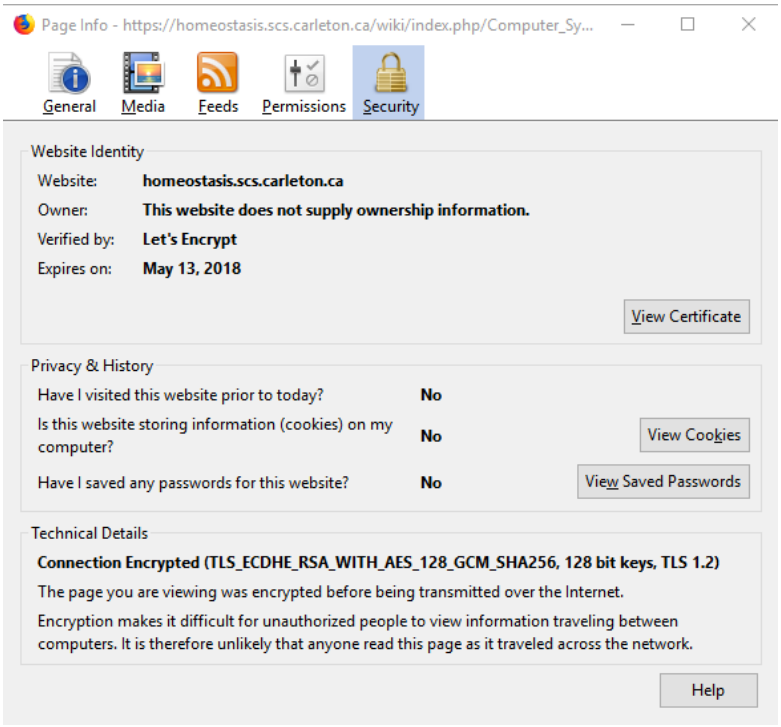

Figure 2: The Course Webpage Certificate

As we can see in Figure 1 beside the URL field is a lock which says *secure* and indicates that there is at least a digital certificate provided for this particular website.

By clicking on the lock icon we can view the contents of the digital certificate (Figure 2).

In this example we know that the authorized authority for this certificate is Let's Encrypt and it uses TLS, AES, RSA and SHA 256 for encryption.

1.2: Relation To Public Keys

Once we have verified that the digital certificate provided is valid and is endorsed by a reputable authority we can then proceed to communicating between the client and the server. As we can see in Figure 2 the digital certificate contains the framework of what type of communication should take place between the client and the server.

The encryption algorithm mentioned in the certificate is a combined TLS_ECDHE_RSA with AES_128_CGM_SHA 256, This means that since TLS is being used for the initial communication it requires the sever and the client to exchange there public keys to form a *handshake* that validates both the user and the server. The RSA part is where the asymmetric public keys are exchanged between the user and the client then the Diffie-Helemn part is where the servers public key is computed with the clients private keys.

With that said we can conclude that the certificate plays a role in specifying what type of algorithm will be used to encrypt the communication between the client and the user. Then once the algorithm is setup both the client and server will exchange there keys either Asymmetrically or Symmetrically.

2.0: SSH Remote Hosts

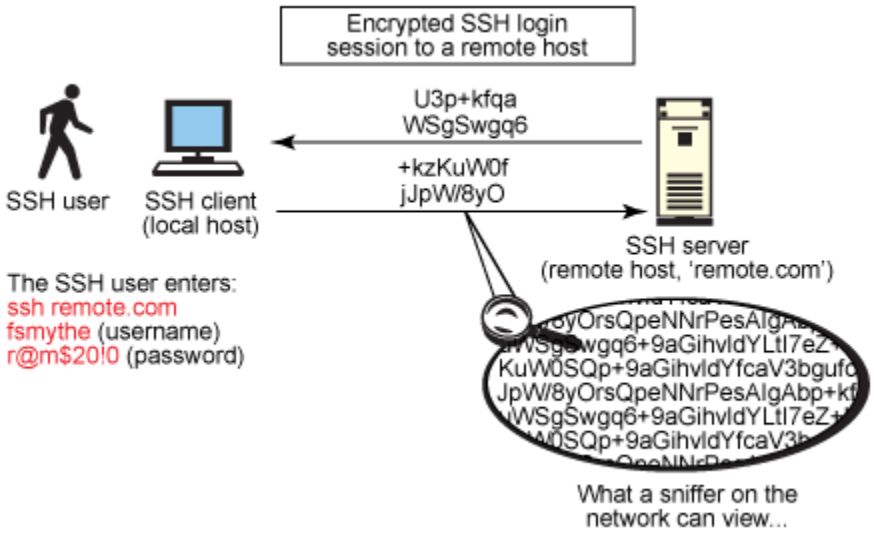

Figure 2: The Secure Shell Protocol

Secure Shells (SSH) operate on a client-server model, this implies that the client must first initialize the communication by calling the SSH Server.

Once the connection is established by the SSH client the SSH server will ask the client to enter there log in credentials to determine if the client is valid.

2.1: SSH Remote Hosts Verification

In order for a secure Shell to create a secure connection between the SSH Server and SSH client the SSH server will need to authenticate the SSH client based on a key exchange between the SSH Server and the SSH Client.

The first step towards authentication requires the SSH client to generate a key pair that includes a public and private key. Once the SSH client has generated a key pair it sends its public key to the SSH server. The SSH server will then generate a new key which we will call (Key X), the SSH server will then encrypt it using the public key sent from the SSH client then send it to the SSH client which then receives it as an encrypted key X.

Using its private key the SSH client decrypts the encrypted key and gets (Key X), it then sends Key X back to the SSH server. Once key X is received by the SSH server it can then authenticate the client as a valid user and can start sending messages between the client and server.

2.2: SSH Future Connections

When the SSH server receives a public key from the SSH client it stores it as an authorized key for future sessions allowing the SSH client to make quick future connections. Since these keys are stored the connection will always stay established between the SSH client and SSH Server. The only requirement would be a user login screen which would reconnect the client and the host.

3.0: Web Browsers and Transport Layer Connections (TLS)

Transport Layer Connections combine the use of symmetric key exchanges with digital certificates. The digital certificates are used to verify the source while the key exchange is then used to encrypt the connection between the server and the client.

The benefits of TLS is If a man-in-the-middle attack were to occur the attacker will be able to view the network traffic between the client and the server, but since the packets are encrypted it would be difficult for the attacker to decrypt what the packets contain.

3.1: TLS Connections Remote Website Verification

Remote website verification is primarily based on the digital certificate the client receives from the server and on what the contents of the certificate contains.

Once the client receives the digital certificate it must either:

A. Approve the certificate since it trusts the authority that endorse the certificate ***or***
B. If the certificate is not currently endorsed by a trusted authority it must be endorsed by an another authority that the client trusts.

Once the digital certificates are authenticated the client will request the following information to the server:

1) The Version of SSL/TSL its running
2) The Ciphersuite it would like to use
3) The Compression Methods it wants to use

The server will then attempt to meet the clients request and send back how best it can meet these set of requirements.

For example the server will check what is the most compatible SSL/TLS version currently installed on the server and try to communicate back to the client with the most compatible version. This could be anywhere from TLS 1.0 – 1.3.

Afterwards the client will then receive the TLS version that's appropriate to both the server and client.

Next the key exchange/ handshake takes place where the server and client will agree to create a common public key while also creating there own separate private keys for the session using the encryption algorithm requested in the ciphersuite part of the digital certificate ie. RSA, ECC or Diffie-Helemen.

Once the public keys are created the client will then encrypt a message to the server to start encrypting all further communications. The server will then decrypt the message using its private keys and validate the clients mac address, thus the validation is now complete.

3.2: TLS Connections Information The Client Receives and How It Is Used

As mentioned in 3.1 the client will receive the digital certificate from the server and must use the information to authenticate if it's a valid source. If it does not come from a valid source then it must check to see if the certificate is valid by another certificate authority it trusts.

The benefit is if its an unknown certificate authority but is endorsed by an authority that the client trusts it will save and add it for future use, Figure 3 lists some of the popular certificate authorities.

In addition to the key exchange mentioned in 3.1 the client will use the common public key received from the server to communicate back to the server using the encryption algorithm in the ciphersuite of the digital certificate.

Figure 3: Popular Certificate Authorities

4.0: Browser Threat Modeling

Browser sandboxing is based on the threat model of if we don't know the code's behaviour and all the possible code inputs/outputs we should assume that the code is malicious.

Since it is almost impossible to fully understand the behaviour of every code the browser should sandbox every website with the assumption that it contains malicious code.

The sandbox does assume that if it detects code trying to call the main(); function it will assume that the code is malicious or if it attempts to gain super-root privileges then it is also malicious.

4.0.1: – Threat Modeling: The Chrome Browser Sandbox

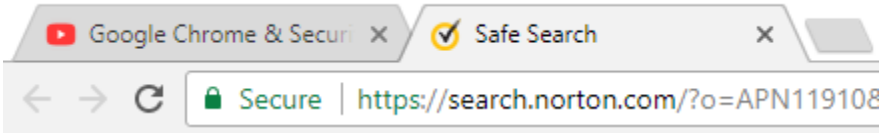

Figure 4: The Chrome Browser With Tabs Acting As Sandboxes

Unlike traditional browsers the Google Chrome browser has sandboxing implemented, each tab was built to contain a separate process and be contained within that tab. The browser also assumes that If the processes within that tab try to gain higher privileges, attempt to execute executable files or call the main function it must mean that the sandbox has been infected and should now be further isolated to prevent it from spreading to the rest of the system.

4.0.2: – Threat Modeling: The FireFox Browser Sandbox

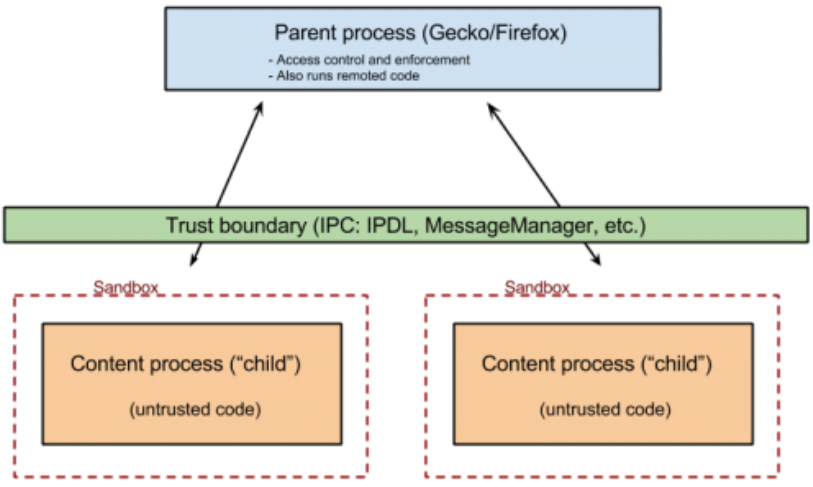

Figure 5: The Firefox Browser With Process Acting As Sandboxes

In contrast to the Chrome browser sandbox Model the Firefox browser sandbox works by sandboxing each child and parent process. The amount of control each child process gets is controlled directly by the parent process. If the sandbox detects the child process attempting to gain more privileges its assumed to be malicious and is isolated.

4.1: The Browser Sandbox Attacks

4.1.1: The Browser Sandbox Attacks: Protection

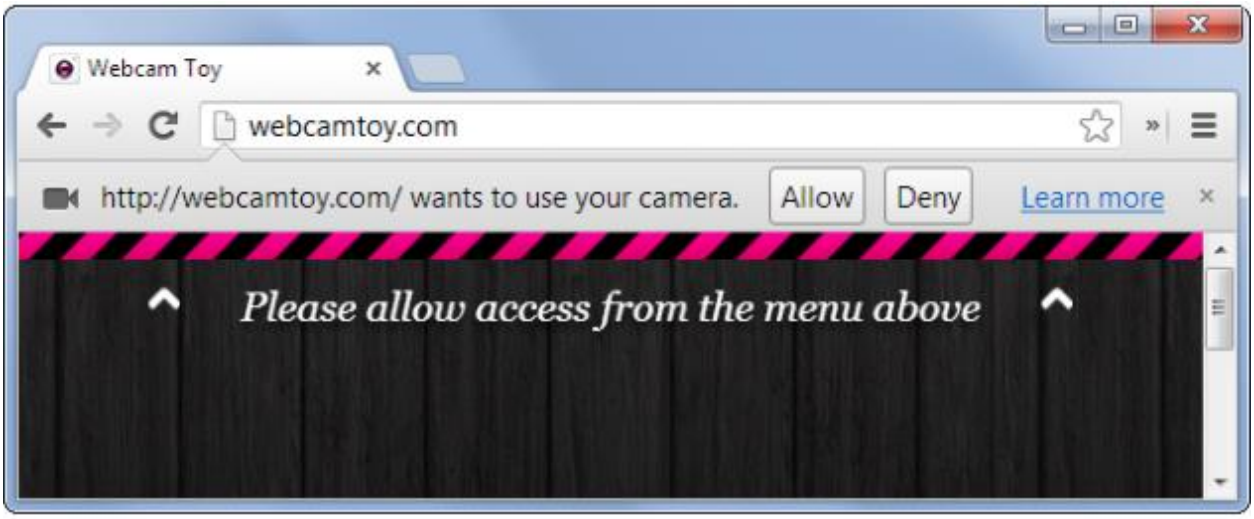

Figure 6: JavaScript Being Blocked

As we can see in Figure 6 the browser sandbox blocked the following JavaScript code from running, thus one of the many protections browser sandbox offers is blocking javascript code from running if the process does not have the privileges to do so.

Plug in Content, PDFs, flash and other browser applications are also equally blocked as it also requires privileges to run.

4.1.2: The Browser Sandbox Attacks: What It Does Not Protect

The browser however does not prevent the user from giving malicious websites access privileges, as we can see in Figure 6 the user could give the malicious website permissions to run and defeat the purpose of having a sandbox.

5.0: Firefox Memory Corruption Vulnerability

For this scenario we assume that the memory corruption vulnerability is a memory buffer overflow which will be susceptible to the stack smashing attack.

Stack smashing occurs when an attacker attempts to cause the memory stack to overflow and replace the existing memory with a different set of data.

Lets say the function/method is about to be executed but before the stack is executed the attacker adds more data into memory.

Since there is more data in the memory the original data becomes the overflow and gets replaced by malicious data that was added by the attacker.

In the case of a memory corruption vulnerability occurring with the Firefox browser we assume that the attacker will attempt to add more bookmarks in the profile folde. Next he/she will attempt a stack smash in the call tree by having the Firefox browser process more bookmarks then is actually allowed.

Once the call tree executes and the stack smash occurs the attacker would have successfully replaced the original data with his/her malicious data.

6.0: Backdoors

Backdoors are typically defined as a method of bypassing authentication of a system, this could be an algorithm or application meant to bypass a users authentication login or In rare cases they can be purposely left by a software administrator to debug or correct future errors in the system.

In the context of computer security a back door would typically come in the form of a malware which is essentially malicious code meant to open a new port(s) that would allow internet traffic to bypasses the firewall or the systems anti virus.

6.1: Backdoors – Hard To Discover

Since a malware needs a way to enter a system it will typical mask its code signature or behaviour as a non malicious application, ie trojan horse virus. Once the malware infects the system it will embed itself into the computers file system or cache itself within the systems memory. Once this occurs it will mimic the code signature of known files on the system and camouflage itself as non malicious.

Since it is able to camouflage itself and mimic the code signature of other files it becomes extremely difficult to detect the malware and where it is in the system.

6.2: Backdoors – Detection and Deletion

Although detecting the actually source of the malware/backdoor can be difficult the signs of a malware infection are very obvious. This could come in the form of a computer lagging in performance, decrease in file storage space capacity, blue screen of death and many others.

When any of the conditions listed above occurs its obvious that a backdoor/malware is taking place. To detect the actual source of the backdoor applications that monitor the internet traffic could be used to monitor the systems internet traffic, this could be the Antivirus or Wireshark.

If discrepancies appear between internet traffic on behalf of the user and various processes attempting to gain internet access then the malware can be detected.

As an example lets say malicious traffic is detected entering a lower level port, none of the users internet traffic matches these requests and because using a lower level port is generally used by the operating system we can tell that a malware is taking place.

Next we procced to check which operating system processes or applications use that port, followed by quarantining the systems files and processes.

We can assume that files that were added after the initial installation are more then likely to contain the malware and proceed to remove them from the system.

7.0: Virtual Machine

Virtual machine is a desktop virtualization application that manages and runs multiple guest operating systems called Virtual Boxes on a single host operating system. The virtualization application isolates each guest operating system and manages the amount of hardware each virtual box is allocated.

7.1: Virtual Machine -Attack Mitigation

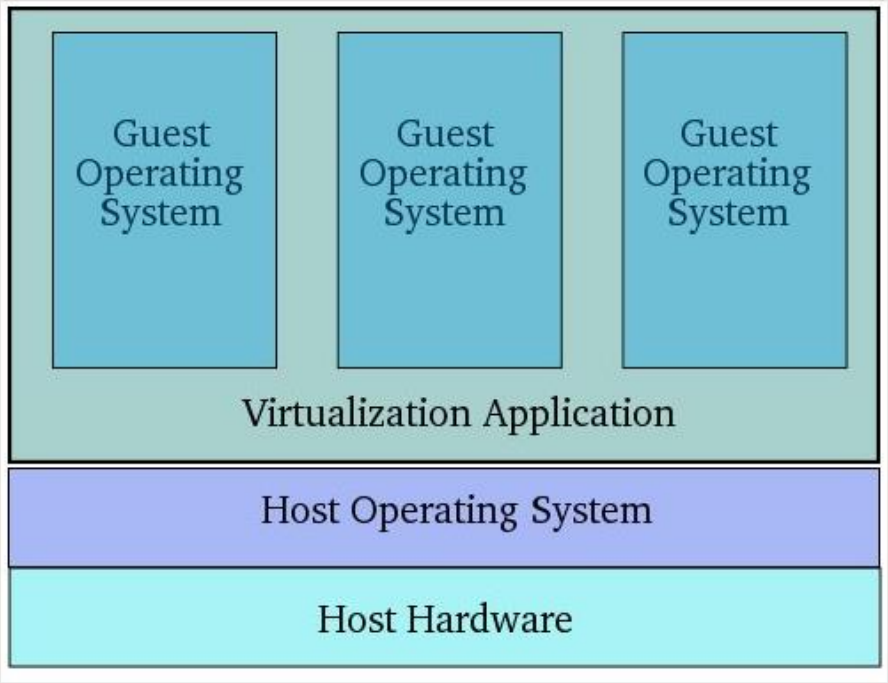

Figure 7: The Virtual Machine Architecture

As we can see in Figure 7 each guest operating system/virtual box is located on a layer that is on top of the host operating system. In addition each virtual box has no direct access to each other or the host operating system which then creates a layer of security based on containment.

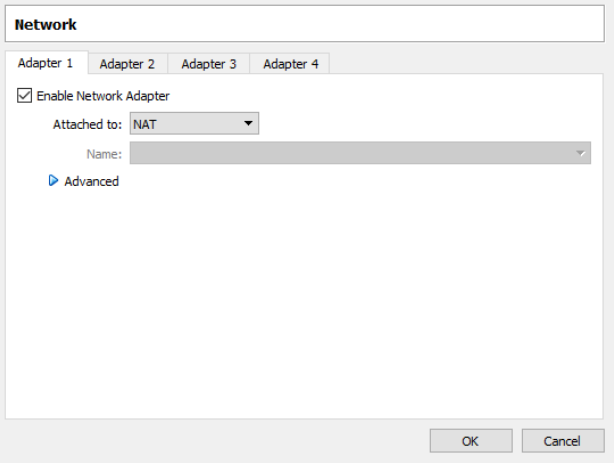

Figure 8: Virtual Box With NAT Settings

Now assuming that each of the Virtual Boxes are connected to the internet using NAT (Network Address Translation) each of the virtual boxes will then become there own separate terminals and have there own unique IPs. Once each of the virtual boxes have there own unique IPs from the viewpoint of a potential attacker each of the terminals would appear as another computer on the network (Figure 9).

Therefore as long as user restricts his/her internet use on one of the virtual boxes, internet traffic will be confined to that box and will not reach the host operating system.

This means threats such as malware/backdoors will be restricted to only the virtual box.

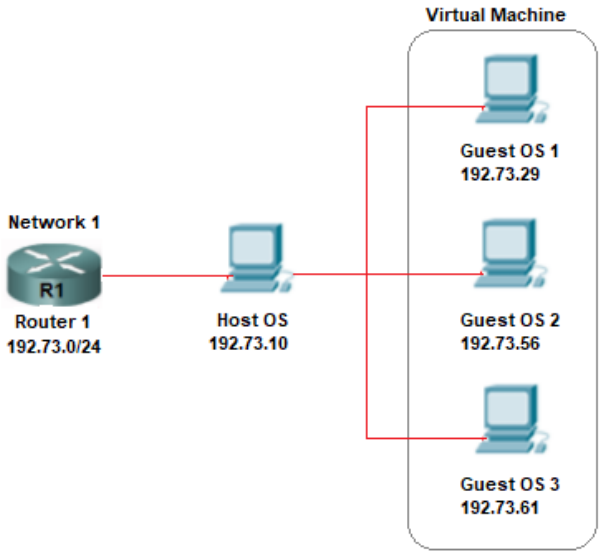

Figure 9: Virtual Machine As Its Own Separate Network

Therefore VirtualBox offers a layer of security based on containment, almost all malicious activity that occurs within the virtual box is stuck within only the virtual box.

7.2: Virtual Machine – New Opportunities For Attackers

What was mentioned in section 7.1 is strictly theoretically on how viruses and malwares should all be contained within a virtual box. Within the last six years more sophisticated malware has started to appear that can defeat virtualization .

According to Infosecurity Magazine 18% of malware [1]can detect if virtualization is in use, this can be achieved by checking the physical hardware of the virtual and where its located.

If the hardware and memory is detected to be outside of the guest operating system the malware will conclude that virtualization must be in use.

Depending on the level of sophistication less sophisticated malware would stop executing since it knows virtualization is being used while more sophisticated malware will change its code behaviour and attempt to defeat detection.

In other situations the malware could wait until its detected and as soon as its been detected release its payload onto the targeted machine.

Some analysts argue there is a potential vulnerability that if one of the virtual boxes is target and becomes infected it could infect the other virtual boxes by attacking the virtual switch.

Even thought each virtual box is independent of the other they are still connected on the same network via a virtual switch. By infected the virtual switch the network essential gets infected and spreads to each box.

This type of vulnerability does exist because current intrusion detection systems are built and designed to operate on traditional networks and not on virtual networks and switches.

[1] Seals, Tara. "Malware No Longer Avoids Virtual Machines", 13 Aug 2013
https://www.infosecurity-magazine.com/news/malware-no-longer-avoids-virtual/

8.0: Access Control

The term access control from a computer security perspective refers to the accessibility of resources and who or what processes are allowed to use that resource. Access control can be broken down into four categories among these categories are mandatory and discretionary access control which will be discussed in the following section.

8.1: Mandatory Access Control

As the name suggests mandatory access control is when the process is granted a level of access for certain resources. If the process wants access to another resource say CPU access the operating system security kernel will check if the processes level of access matches the level required for that resource.

Since Mandatory Access Control has the highest amount of control among the four categories its advantage is that it offers the highest level of security available.

The disadvantages to implementing mandatory access control is that in order to maintain such a high level of security meticulous checking of the resources and the amount of resources needed by each process is required. Failing to monitor and track each process and what there resource needs could create a processor deadlock condition.

8.2: Discretionary Access Control

The main advantage of discretionary access control is assume that a process or user has rights to a specific program/application but hasn't been granted the rights to run the resource.

Instead of blocking the program and resource because neither has the adequate access level, Discretionary Access Control will allow the process to gain rights to the resource in order to run the application. In other words discretionary control can transfer resource rights to other process that may need them.

The disadvantage of discretionary access control is malwares such as a trojan can mimic other processes that need access to certain resources, then the trojan can cause damage to a system after it gains rights to run resources that it shouldn't have been given right to.

9.0: Operating Systems

9.1: Integrity and Authentic Of Initial Installation

The traditional method of checking for data integrity of any code is to use chkflags() which compares the difference of two files. The suggested way for Ubuntu by the publisher is to use the SHA256SUMS.gpg file and compare it against the sha256sum of the ISO file we used to install the OS.

If there are discrepancies we would know that the installation is not authentic.

9.2: Verification Of Software Updates

Similar to how installation files are verified for authenticity by using checksums we can verify the updates using debsums which is an MD5 check sum that checks the systems.

```
$ sudo debsums
```

Figure 10: Typing In The Debsums Command

```
/usr/bin/a11y-profile-manager-indicator                                  OK
/usr/share/doc/a11y-profile-manager-indicator/copyright                  OK
/usr/share/man/man1/a11y-profile-manager-indicator.1.gz                  OK
/usr/share/accounts/providers/facebook.provider                          OK
/usr/share/accounts/qml-plugins/facebook/Main.qml                        OK
/usr/share/accounts/services/facebook-microblog.service                  OK
/usr/share/accounts/services/facebook-sharing.service                    OK
/usr/share/doc/account-plugin-facebook/copyright                         OK
/usr/share/accounts/providers/flickr.provider                            OK
/usr/share/accounts/qml-plugins/flickr/Main.qml                          OK
/usr/share/accounts/services/flickr-microblog.service                    OK
/usr/share/accounts/services/flickr-sharing.service                      OK
/usr/share/doc/account-plugin-flickr/copyright                           OK
/usr/share/accounts/providers/google.provider                            OK
/usr/share/accounts/qml-plugins/google/Main.qml                          OK
/usr/share/accounts/services/google-drive.service                        OK
/usr/share/accounts/services/google-im.service                           OK
/usr/share/accounts/services/picasa.service                              OK
/usr/share/doc/account-plugin-google/copyright                           OK
/lib/systemd/system/accounts-daemon.service                              OK
/usr/lib/accountsservice/accounts-daemon                                 OK
/usr/share/dbus-1/interfaces/org.freedesktop.Accounts.User.xml           OK
/usr/share/dbus-1/interfaces/org.freedesktop.Accounts.xml                OK
/usr/share/dbus-1/system-services/org.freedesktop.Accounts.service       OK
/usr/share/doc/accountsservice/README                                    OK
/usr/share/doc/accountsservice/TODO                                      OK
```

Figure 11: The Results Of The MD5 Scan

As we can see the MD5 Scan shows the results of the updated services, if it reads OK we can tell the updates are authentic.

9.3: Third Party OS Changes

For the Android operating system several third party custom firmwares do exist that modify the android OS these include AOKP, CopperheadOS and Android Oreo.

Each of the three listed works by customizing the ROM distribution on the Android OS.

9.4: What OS Can Do To Protect Themselves From Modification

To counter the threat of OS modification on Unix based systems such as Linux and Mac OS X, UNIX based systems have now started to implement a Systems Integrity Protection called rootless, this means that processes with higher privileges can not gain access to higher level kernel functions and make modifications to the OS.

10.0: Bibliography

Section 1.0

Stack Exchange. “How Do Digital Certificates Work”, 14 Jun 2012, https://crypto.stackexchange.com/questions/2893/how-do-digital-certificates-work-and-why-is-it-not-possible-to-reverse-engineer

Section 2.0

SSH Communications Security. “SSH Protocol”, 19 Aug 2017, https://www.ssh.com/ssh/protocol/

Ellingwood, Justin. “Understanding The SSH Encryption and Connection Process” 22 October 2014. https://www.digitalocean.com/community/tutorials/understanding-the-ssh-encryption-and-connection-process

Bifumes Webnologies. “How SSH Key Works”, 24 Jan 2018, https://www.youtube.com/watch?v=y2SWzw9D4RA&t=340s

Section 3.0

Stack Exchange. “Details Of TLS Certificate”, 8 Oct 16, https://security.stackexchange.com/questions/139176/details-of-tls-certificate-verification

Stack Exchange. “How Does SSL/TLS Work”, 21 Dec 14, https://security.stackexchange.com/questions/20803/how-does-ssl-tls-work

Stack Exchange. “How Does The Symmetric Key Get Exchanged in SSL/TLS Handshake”, 25 Jul 16, https://security.stackexchange.com/questions/130938/how-does-the-symmetric-key-get-exchanged-in-ssl-tls-handshake

Dierks, T. “The Transport Layer Security (TLS) Protocol Version 1.2”, August 2008, https://tools.ietf.org/html/rfc5246

Section 4.0

Quora. “What Is Google Sandbox”, 5 Feb 2016, https://www.quora.com/What-is-Google-Sandbox-1

Mozillia. “Security/Sandbox”, 15 Feb 2018, https://wiki.mozilla.org/Security/Sandbox

Chromium. “Sandbox”, https://chromium.googlesource.com/chromium/src/+/b4730a0c2773d8f6728946013eb812c6d3975bec/docs/design/sandbox.md

Hoffman, Chris. How-To Geek. “Sandboxes Explained: How They’re Already Protecting You and How To Sandbox Any Program”, 2 Aug 2013. https://www.howtogeek.com/169139/sandboxes-explained-how-theyre-already-protecting-you-and-how-to-sandbox-any-program/

Section 5.0

Mozillia, “Firefox Developer Tools: Call Tree”. https://developer.mozilla.org/en-US/docs/Tools/Performance/Call_Tree

Rouse, Margaret. TechTarget. “Stack Smashing”, http://searchsecurity.techtarget.com/definition/stack-smashing

Section 6.0

Trendmicro. “Backdoor Attacks: How They Work and How To Protect Against Them” 7 Jan 2015. https://blog.trendmicro.com/backdoor-attacks-work-protect/

Kiguolis, Linas. “How To Remove Backdoors”, 11 May 2017, https://www.2-spyware.com/backdoors-removal

Section 7.0

Seals, Tara. “Malware No Longer Avoids Virtual Machines”, 13 Aug 2014, https://www.infosecurity-magazine.com/news/malware-no-longer-avoids-virtual/

Rankin, Bert “Evasive Malware Detects and Defeats Virtual Machine Analysis”, 24 Oct 2014, https://www.lastline.com/blog/evasive-malware-detects-and-defeats-virtual-machine-analysis/

Infosec Institute, “How Malware Detects Virtualized Environment (And Its Countermeasures)”, 11 Feb 2016, http://resources.infosecinstitute.com/how-malware-detects-virtualized-environment-and-its-countermeasures-an-overview/#gref

“6.4. Network Address Translation Service”, https://www.virtualbox.org/manual/ch06.html#network_nat_service

Stack Exchange. “Does A Virtual Machine Stop Malware From Doing Harm”, 25 Oct 17, https://security.stackexchange.com/questions/9011/does-a-virtual-machine-stop-malware-from-doing-harm

Virtuatopia. “An Overview Of VirtualBox2”, 4 March 2009, http://www.virtuatopia.com/index.php/An_Overview_of_VirtualBox_2

VMware vSpher4- Esx and vCenter Server, “Security and Virtual Machines”. https://pubs.vmware.com/vsphere-4-esx-vcenter/index.jsp#com.vmware.vsphere.server_configclassic.doc_40/esx_server_config/security_for_esx_systems/c_security_and_virtual_machines.html

Section 8.0

Techopedia. “Discretionary Access Control”, https://www.techopedia.com/definition/229/discretionary-access-control-dac

Turnbull, James. TechTarget, “DAC and MAC Safety”, http://searchdatacenter.techtarget.com/answer/DAC-and-MAC-safety

Rouse, Margaret. “Access Control” Jun 2014. http://searchsecurity.techtarget.com/definition/access-control

Section 9.0

Ubuntu. “How To Verify Your Ubuntu Download”, https://tutorials.ubuntu.com/tutorial/tutorial-how-to-verify-ubuntu?_ga=2.72152347.945106664.1520149126-552941175.1520149126#0

Kill, Aaron. “How To Check MD5 Sums Installed Packages In Debian/Ubuntu Linux”10 Oct 2016, https://www.tecmint.com/check-verify-md5sum-packages-files-in-linux/

Newell, Gary. “How To Validate The MD5 Checksum Of A File”, 12 Feb 2018, https://www.lifewire.com/validate-md5-checksum-file-4037391

Palstsiuk, Viktar. “Updating The Firmware Of Linux-Based Devices”, 24 Apr 2013, http://www.linuxjournal.com/content/updating-firmware-linux-based-devices

Watson, JA. “Verifying The Integrity Of Linux ISO Images” 10 Mar 2016, http://www.zdnet.com/article/verifying-the-integrity-of-linux-iso-images/

Nanni, Dan. “How To Verify The Authenticity and Integrity Of A Downloaded File On Linux”, 16 Oct 2014, http://xmodulo.com/verify-authenticity-integrity-downloaded-file.html

LinuxForever. “How To Verify The Integrity and Authenticity Of Linux Mint ISO File”, 3 Jun 2017, http://linuxforever.info/2017/06/03/how-to-verify-the-integrity-and-authenticity-of-linux-mint-iso-file/

Hoffman, Chris. How-To Geek, “How To Disable System Integrity Protection On A Max (and Why You Shouldn’t)”, 1 May 2017, https://www.howtogeek.com/230424/how-to-disable-system-integrity-protection-on-a-mac-and-why-you-shouldnt/