Systems Security

Learning Outcomes

- Identify threats to e-commerce resources
- Internet crimes
- How to maintain Confidentiality, integrity and availability
- Technical and non technical attacks
- Security mechanisms

    a. Cryptographic techniques
    b. Firewalls

Threats to System/Network Security

There are four primary classes of threats to system/network security

Unstructured Threats

- consist mostly of inexperienced individuals using easily available tools such as shell scripts and password crackers

Structured Threats

- from hackers who are highly motivated and technically competent. They know and are aware of system vulnerabilities. They can understand and develop exploit code and scripts. They use sophisticated hacking techniques to penetrate and compromise systems

External Threats

– these arise from individuals or organizations working outside the company, institution or organization. They do not have authorized access to the computer systems or network(s). They work their way into a network mainly through an Internet connection

Internal Threats

– these occur when someone has authorized access to the network with either an account on a server or physical access to the network

Internet crimes

- **SQL Injection Attacks**

❖ This involves the attackers finding flaws in websites that have databases running behind them.

❖ A poorly validated input field in a web input form may allow an attacker to insert additional SQL instructions which may then be passed directly into the backend database.

- **Cross Site Scripting**

❖ In its simplest form, it’s a process that can occur anywhere a web application uses input from a malicious user to generate output without validating or encoding the input.

❖ During a Cross Site Scripting attack, a malicious source sends a script that is executed by the end user’s browser. It allows attackers to embed code from one webpage into another webpage by changing its HTML code.

❖ It’s been used to deface websites, conduct phishing attacks, or it can take over a user’s browser and force them to execute commands they’re unaware of.

❖ Cross Site Scripting attacks usually come in the form of JavaScript

Cont..

- **Poisoned search engine**

❖ The attackers create numerous websites for well known keywords and use these sites to feature high up on web searches.

❖ When a user searches for a particular name or subject and clicks on a poisoned link, he or she is taken to a fake website where they are told to either download software to continue or else malware is downloaded while they are browsing the content.

- **Malicious Advertisements (Malvertising)**

❖ Malvertising is the use of online, malicious advertisements to spread malware and compromise systems.

❖ Generally, this occurs through the injection of unwanted or malicious code into ads.
❖ Many websites today display advertisements hosted by third-party advertising sites
❖ The volume of ads published automatically makes detection difficult

❖ Random appearances further compound the detection

Cont..

- **Web shells**

❖ A web shell is an executable code running on a server that gives an attacker remote access to functions of the server.

❖ A web shell can also be seen as a type of Remote Access Tool (RAT) or backdoor Trojan file

❖ Web shells are installed on a web server through a compromise of some kind.

❖ The compromise could be through a legitimate web application on the server using techniques like SQL injection, Remote File Inclusion, unvalidated file upload feature or through a valid user's stolen credentials.

❖ When the shell is installed, it will have the same permissions and abilities as the user who put it on the server.

❖ The shell may be a full featured administrative GUI or as simple as a single line of code that simply takes commands through a browser's URL field and passes them on to the back-end server.

- **Malicious File Execution**

❖ When developers program applications to use input files provided by the user and the attacker is the one entering the file, a malicious file is executed unknowingly, thus we have malicious file execution.

❖ Malicious file execution attacks can occur anytime the application accepts filenames or files from users.

❖ When these files are executed, they can be used to do just about anything from stealing data to taking over the entire system.

- **Denial-of-Service (DoS)**

❖ An attacker causes the web server to be unavailable.

❖ To successfully launch a DoS attack, an attacker frequently requests many pages from your website.

❖ Your server is unable to handle these many requests at a time and as a result becomes very slow

❖ A common form of DoS attack is the DDoS (Distributed DoS)

Denial of service cont..

In DDoS attacks, hackers use lots of computers to frequently requests many pages from your website.

❖ As a result of DDoS attacks:

- Users cannot get to your site.

-The server may crash and lose or corrupt important data.

-All the bandwidth used by the attackers may cost you a lot of cash.

- **Third-party add-ons**

❖ The majority of websites require the use of third-party add-ons such as Adobe Flash player and Acrobat Reader.

❖ Both of these widely used products have become a favourite target for cybercriminals.

❖ These third-party add-ons are used to push users to other websites that have been compromised.

Topic Review Questions

- 1) Why do hackers target websites/systems more than any other internet resource? (3mks)
- 2) How do you differentiate genuine from fake/phishing websites? (3mks)
- 3) What are signs that a website has been hacked? (3mks)
- 4) As a web/systems administrator, what can you do to recover from a hacking incidence? (3mks)

Mitigating Web/Systems Attacks

- **Protecting websites against cross-site scripting**

❖ Validate the users input against what is expected

❖ Encode user supplied output

❖ After code development, inspect your code with a scan.

- **Protecting websites against SQL Injection**

❖ You can put tight constraints on user inputs.

❖ However, the best method of preventing SQL injection is to avoid the use of dynamically generated SQL in your code. Instead, use stored or canned procedures.

❖ And then again, run a scan to make sure your application is not vulnerable to SQL injections.

- **Protecting websites/systems against Malicious File Execution**

❖ Strongly validate user input using "accept known good" as a strategy, or isolate incoming files and check their legitimacy before executing them.

❖ Disable some default PHP functions/commands.

Pro-tip: visit the OWASP website to see what php commands to disable.

- **Protecting against Fake web URLs**

❖Learn how to recognize a fake URL and how to set your web browser's security settings to protect you against such sites.

❖Turn off automatic pop ups and disable ads and Javascript in your browser for added protection.

- **Protecting against Web shells**

❖ Periodically do a search of all files in the web root hierarchy looking for the functions that a shell depends on, such as the eval(), passthru(), exec() and system() if running PHP or the equivalent in the supported languages.

❖ Logs can also indicate there is a shell on a server and in use. Watch for unusual requests to files when the requests do not correlate or don't make sense by protocol e.g. a PDF or JPG file being called with GET parameters could be an indication the file extension is not accurate and that it is actually a web shell.

❖ With WAF rules turned on and watching for characteristics of requests and responses with a shell, you can get an indication of a web shell in use and determine its location on the server

- **Protecting against Adware, spyware, scareware, and viruses**

❖ Avoid downloading free software unless you are certain it's from a reputable company.

❖ Be careful of sites that tell you to install a new "plug-in" or "media player" to continue.

❖ Another way criminals try to snag users is by offering "malware protection" through a message that pops up saying a virus has already been detected on the user's computer (hence the term "scareware"). Instead of fixing the "problem," users end up downloading a virus laded piece of software.

❖ Install virus protection software and make sure your firewall is turned on for added protection.

General Webserver/systems Hardening Tips

❖ System hardening means reducing the attack surface thereby making it more difficult for a malicious hacker to attack.

❖ To harden your web server:

❑ Remove all unnecessary web server modules. A lot of web servers by default come with several modules that introduce security risks.

❑ Modify the default configuration settings. For example, a lot of web servers support old SSL/TLS protocols in their default settings. This means that your server is vulnerable to attacks such as BEAST or POODLE.

❑ Turn on additional protection for web applications. For example, introduce a Content Security Policy (CSP).

❑ Install and run a web application firewall (WAF). Most web servers support the open-source ModSecurity firewall.

❑ If possible, either patch server software to the latest version automatically or turn on notifications for manual patching.

❑ Regularly scan all your web applications using a web vulnerability scanner. Eliminate all vulnerabilities as early as possible.

❖Hardening should be a continuous (never ending) process. This is because new vulnerabilities are discovered every day

❖You should perform regular system hardening check-ups to make sure that your security configuration is up to date, all the security measures are still in place, and there are no new threats to your information security.

Apache Webserver Hardening Tips

- **Securing the Apache web server**

❖ The Apache web server is one of the most popular web servers available for both Windows and Linux/UNIX.

❖ According to W3Techs, Apache is the most popular web server powering 47% of the websites with a known web server as of 2021.

❖ Due to its popularity, Apache is a prime target for hackers. As such, extra security measures must be taken after initial setup.

- **Apache web server hardening tips**

❖ Hide Apache version and OS identity from errors: When you install Apache, it displays the version of your Apache web server with the operating system name in the errors. It also shows the information about Apache modules installed. You should always configure to turn these default features off.

❖ Disable Directory Listing: By default, Apache lists all the content of Document root directory in the absence of index file.

❖ Keep updating Apache regularly: The Apache developer community is continuously working on security issues and releasing its updated version with new security options. It is always recommended to use the latest version of Apache as your web server.

❖ Disable unnecessary modules: It’s always good to reduce the attack surface by disabling all those modules that are not currently in use.

❖ Run Apache as separate User and Group: For security reasons, it is recommended to run Apache in its own non-privileged account

❖ Use "Allow" and "Deny" options in the Apache config file to restrict access to directories.

❖ Limit Request Size: by default, Apache has no limit on the total size of the HTTP request. When you allow large requests on a web server, it’s possible that you could be a victim of Denial of service attacks.

❖ Minimize the chances for DDOS attacks: this can be achieved by setting limits on the Apache web server e.g.

TimeOut: This directive allows you to set the amount of time the server will wait for certain events to complete before it fails.

MaxClients: This directive allows you to set the limit on connections that will be served simultaneously.
KeepAliveTimeout: It’s the amount of time the server will wait for a subsequent request before closing the connection

❖ Enable Apache logging: it is wise to enable Apache logging, because it provides more information, such as the commands entered by users that have interacted with your Web server.

❖ Securing Apache with SSL Certificates: you can secure all the communication in an encrypted manner over the Internet with SSL certificate. Apache sends all this information in encrypted text.

❖ Use ModSecurity Module (modsec) to secure Apache: Modsec works as a firewall for our web applications and allows us to monitor traffic on a real time basis. It also helps us to protect our websites or web server from brute force attacks.

Modsecurity WAF

- **Modsec (ModSecurity)**

❖ ModSecurity is an open-source module that works as a web application firewall (WAF).

❖ It is a set of rules with regular expressions that helps to instantly ex-filtrate the commonly known exploits

❖ Modsecurity supplies an array of request filtering and other security features to the web server.

❖ As such, it provides different functionalities including filtering, server identity masking, and null-byte attack prevention.

❖ This module also lets you perform real-time traffic monitoring.

❖ ModSecurity protect against a multitude of attacks including DDOS (distributed denial of service) attacks. It also obstructs the processing of invalid data (code injection attacks) to reinforce and nourish server's security.

❖ You can also temporarily use modsec to protect against certain attacks like SQL Injection and Cross-site Scripting until vulnerabilities are fixed by the developer.

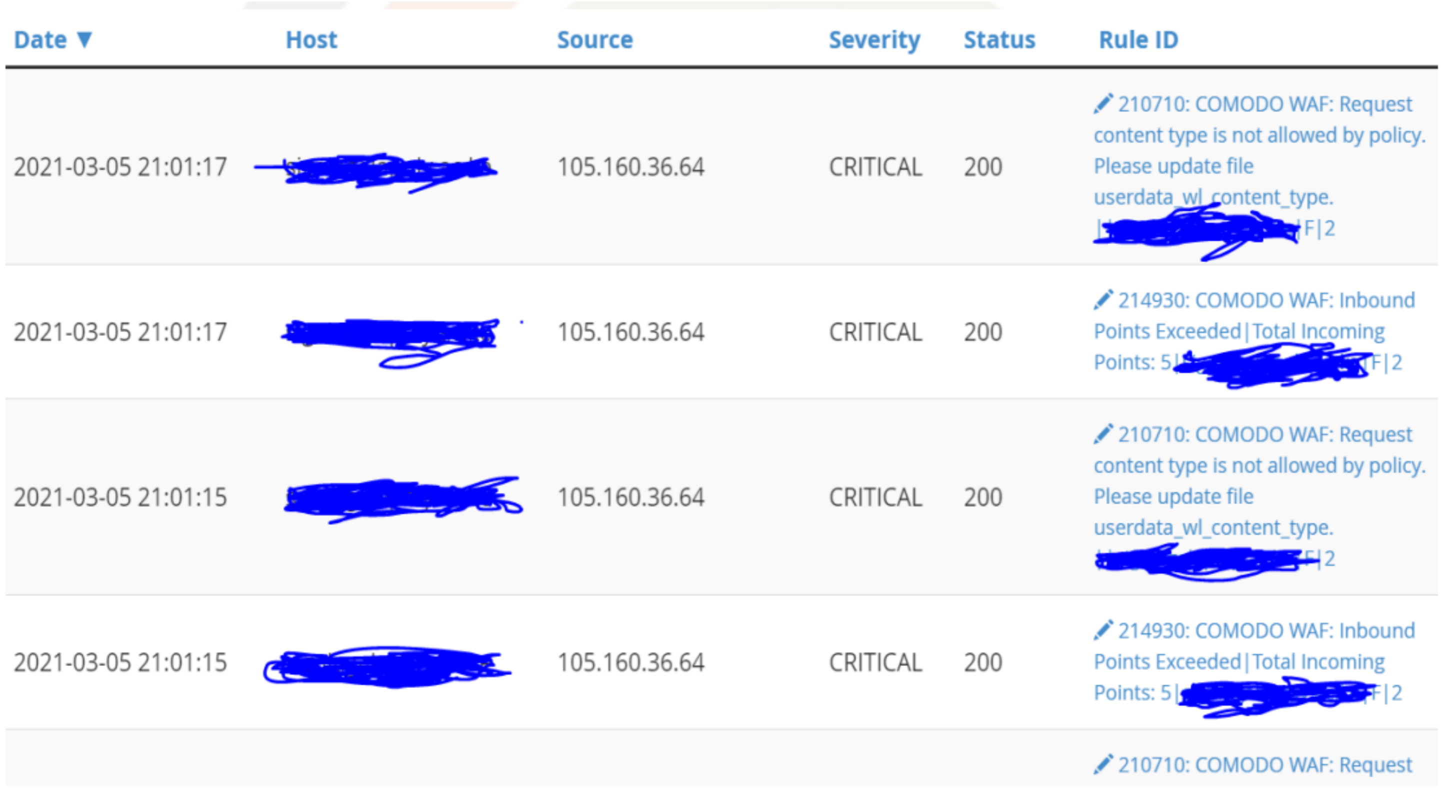
There are free updated rules from

1.**OWASP**: the OWASP ModSecurity Core Rule Set (CRS) is a set of generic attack detection rules for use with ModSecurity or compatible web application firewalls. url: https://owasp.org/www-project-modsecurity-corerule-set/

2.**COMODO**: Comodo ModSecurity is the best Web Application Firewall for web apps and websites running on Apache/Linux web-servers. url: https://modsecurity.comodo.com/

NB:You can also write your own custom modsec rules

Sample Modsec Logs (Hit list)



Date ▼	Host	Source	Severity	Status	Rule ID
2021-03-05 21:01:17	[redacted]	105.160.36.64	CRITICAL	200	✎ 210710: COMODO WAF: Request content type is not allowed by policy. Please update file userdata_wl_content_type. [redacted]F|2
2021-03-05 21:01:17	[redacted]	105.160.36.64	CRITICAL	200	✎ 214930: COMODO WAF: Inbound Points Exceeded|Total Incoming Points: 5|[redacted]F|2
2021-03-05 21:01:15	[redacted]	105.160.36.64	CRITICAL	200	✎ 210710: COMODO WAF: Request content type is not allowed by policy. Please update file userdata_wl_content_type. [redacted]|2
2021-03-05 21:01:15	[redacted]	105.160.36.64	CRITICAL	200	✎ 214930: COMODO WAF: Inbound Points Exceeded|Total Incoming Points: 5|[redacted]F|2
					✎ 210710: COMODO WAF: Request

Prepared by Salome Mwangi

Modsecurity Rules Review Questions

1. What do you understand by web server hardening? (2mks)

2. Explain the benefits of protecting your institutional website (3mks)

3. What are the advantages of using modsec? (3mks)

4. Discuss any three issues associated with the use of modsec WAF to secure the Apache web server (3mks)

References

https://www.acunetix.com/blog/articles/10-tipssecure-apache-installation/
http://www.modsecurity.org/
https://www.supportpro.com/blog/mod_securityintro/
http://www.inmotionhosting.com/support/website/ modsecurity/what-is-modsecurity-and-why-is-itimportant https://modsecurity.comodo.com/
https://owasp.org/www-project-modsecurity-corerule-set