Operating System Detection:

Operating System (OS) detection is the process of identifying the operating system running on a networked device. This is a critical aspect of network scanning and reconnaissance, as knowing the OS can help in understanding the potential vulnerabilities and exploits that may be applicable to that device.

How is OS Detection Done?

OS detection is typically done using techniques that analyze the characteristics of network traffic and responses from the target system. There are two main types of OS detection:

1. **Active OS Detection**:
   - Sends specially crafted packets to the target and analyzes the responses.
   - Commonly uses TCP/IP stack fingerprinting, where differences in how OSes implement the TCP/IP stack are used to identify the OS.
   - Tools like Nmap send a series of packets and analyze details such as TTL, window size, and flags in the responses.
2. **Passive OS Detection**:
   - Analyzes existing traffic between the target and other systems without sending any packets directly.
   - Uses similar principles to active detection but relies on observing traffic that is already being generated by the target system.
   - Tools like p0f capture and analyze packets to infer the OS based on the same TCP/IP stack differences.

Examples of OS Detection Tools

1. **Nmap**:
   - Nmap is a widely used network scanning tool that includes OS detection capabilities. It uses a combination of active and passive techniques to identify the OS.
   - **Example Command**:

     ```
     nmap -O 192.168.1.1
     ```

   - This command will perform an OS detection scan on the target IP address.
2. **p0f**:
   - p0f is a passive OS fingerprinting tool that identifies the OS based on observed network traffic.
   - **Example Command**:

     ```
     p0f -i eth0
     ```

   - This command will analyze traffic on the `eth0` network interface to passively identify the OS of devices communicating on the network.

3. **Xprobe2**:
    - Xprobe2 is another tool for active OS detection, using various techniques to identify the target OS.
    - **Example Command**:

      ```
      xprobe2 -v 192.168.1.1
      ```

    - This command will perform an OS detection scan on the target IP address with verbose output.

Websites for OS Detection

Several websites and online tools offer OS detection services. These tools can be useful for quickly identifying the operating system of a target device without needing to install any software locally. Here are some popular websites for OS detection:

1. **CentralOps.net**:
    - **URL**: CentralOps.net
    - **Description**: CentralOps.net offers a suite of online tools for network reconnaissance, including OS detection. Their services include Whois, DNS, and traceroute, providing a comprehensive set of utilities for network analysis.
2. **Ping.eu**:
    - **URL**: Ping.eu
    - **Description**: Ping.eu provides a variety of network tools such as ping, traceroute, and port checking. While primarily focused on connectivity testing, it also offers basic OS detection capabilities.
3. **MXToolbox**:
    - **URL**: MXToolbox
    - **Description**: MXToolbox is a powerful tool for diagnosing email and DNS-related issues. It includes OS detection as part of its suite of diagnostic tools.
4. **IP-Details.com**:
    - **URL**: IP-Details.com
    - **Description**: IP-Details.com provides IP lookup and OS detection services. It offers information about the IP address, including the likely operating system running on the host.
5. **IPFingerPrints.com**:
    - **URL**: IPFingerPrints.com
    - **Description**: IPFingerPrints.com offers an array of tools for network reconnaissance, including OS detection, port scanning, and traceroute.

Automated Scanning Workflows

Automated scanning workflows refer to the use of software tools and scripts to perform network, vulnerability, and compliance scans without manual intervention. These workflows help streamline the process of identifying security weaknesses, ensuring continuous monitoring, and maintaining the security posture of an organization. By automating repetitive tasks, security teams can focus on analyzing results and responding to threats more effectively.

Techniques of Automated Scanning Workflows

1. **Scheduled Scanning**: Configuring scans to run at regular intervals, such as daily, weekly, or monthly. This ensures consistent monitoring without manual initiation.
2. **Triggered Scanning**: Initiating scans based on specific events or conditions, such as the detection of a new device on the network, changes in network configuration, or updates to software and hardware.
3. **Continuous Scanning**: Implementing tools that provide real-time or near-real-time scanning capabilities to detect vulnerabilities and threats as they emerge.
4. **Integration with CI/CD Pipelines**: Embedding security scans into the Continuous Integration/Continuous Deployment (CI/CD) workflows to detect vulnerabilities in code before it is deployed to production environments.
5. **Policy-Based Scanning**: Defining and enforcing security policies that automatically trigger scans and checks to ensure compliance with organizational or regulatory standards.

Tools for Automated Scanning Workflows

1. **Nessus**:
   - **Description**: A widely used vulnerability scanner that can be automated to run scheduled scans and generate reports.
   - **Features**: Vulnerability detection, compliance checks, integration with SIEM tools.
   - **Automation Example**: Scheduling weekly scans and configuring email alerts for critical vulnerabilities.
2. **OpenVAS**:
   - **Description**: An open-source vulnerability scanner that offers comprehensive scanning capabilities.
   - **Features**: Automated scanning, customizable scan configurations, detailed reporting.
   - **Automation Example**: Setting up cron jobs to run OpenVAS scans at specific times and exporting results to a centralized database.
3. **Qualys**:
   - **Description**: A cloud-based security and compliance suite offering continuous vulnerability scanning and monitoring.
   - **Features**: Automated scans, asset discovery, integration with other security tools.
   - **Automation Example**: Continuous scanning of all network assets with automated alerting and reporting.

4. **Nmap**:
    - **Description**: A powerful network scanning tool used for network discovery and security auditing.
    - **Features**: Host discovery, port scanning, OS detection, scripting capabilities.
    - **Automation Example**: Using Nmap Scripting Engine (NSE) scripts for automated scanning and reporting.
5. **Burp Suite**:
    - **Description**: A comprehensive web vulnerability scanner and testing platform.
    - **Features**: Automated scanning, manual testing tools, integration with CI/CD pipelines.
    - **Automation Example**: Running automated scans of web applications and integrating with CI/CD tools to identify vulnerabilities during the development process.
6. **ZAP (OWASP Zed Attack Proxy)**:
    - **Description**: An open-source web application security scanner.
    - **Features**: Automated and manual testing, API testing, integration with CI/CD.
    - **Automation Example**: Configuring automated scans for web applications with CI/CD integration for continuous security testing.
7. **Metasploit**:
    - **Description**: A penetration testing framework that includes automated exploitation and vulnerability validation tools.
    - **Features**: Exploitation, vulnerability scanning, integration with other tools.
    - **Automation Example**: Using Metasploit Pro to schedule automated vulnerability scans and exploit validation.
8. **Ansible**:
    - **Description**: An IT automation tool that can be used to automate security tasks.
    - **Features**: Automation of repetitive tasks, configuration management, orchestration.
    - **Automation Example**: Writing playbooks to automate security scans and apply patches across multiple systems.

Scanning for Web Applications

Web application scanning involves identifying vulnerabilities in web applications before attackers can exploit them. The goal is to ensure the security of web applications by detecting issues like SQL injection, cross-site scripting (XSS), and other common vulnerabilities.

Techniques for Web Application Scanning

1. **Static Application Security Testing (SAST)**:
    - Analyzes source code or compiled versions of code to identify security vulnerabilities.
    - Example: Checkmarx, Fortify, Veracode.
2. **Dynamic Application Security Testing (DAST)**:
    - Tests the application in its running state to identify vulnerabilities that could be exploited.
    - Example: OWASP ZAP, Burp Suite, Acunetix.
3. **Interactive Application Security Testing (IAST)**:

- Combines elements of SAST and DAST by testing applications in real-time during execution.
- Example: Contrast Security, Seeker.

4. **Software Composition Analysis (SCA)**:
   - Analyzes the components and libraries used in an application to identify known vulnerabilities.
   - Example: Snyk, Black Duck, WhiteSource.
5. **Penetration Testing**:
   - Manual testing by security professionals to simulate attacks on the application.
   - Example: Metasploit, Core Impact.
6. **Fuzz Testing**:
   - Inputs random data into the application to find vulnerabilities.
   - Example: Peach Fuzzer, AFL (American Fuzzy Lop).

Tools for Web Application Scanning

1. **OWASP ZAP (Zed Attack Proxy)**:
   - An open-source DAST tool.
   - Features: Automated scanners, passive scanning, and a set of tools for finding security vulnerabilities manually.
   - Example Use: Scanning a web application for common vulnerabilities.
2. **Burp Suite**:
   - A comprehensive platform for web application security testing.
   - Features: Intruder, Scanner, Repeater, and Decoder for various security testing tasks.
   - Example Use: Automating scans and manually testing web applications for security flaws.
3. **Acunetix**:
   - A commercial web vulnerability scanner.
   - Features: Automated crawling and scanning, advanced manual tools, and integrations with CI/CD pipelines.
   - Example Use: Detecting SQL injection, XSS, and other vulnerabilities in web applications.
4. **Netsparker**:
   - A web application security scanner.
   - Features: Automation of scanning, proof-based scanning to eliminate false positives, and extensive reporting.
   - Example Use: Regular scanning of web applications to ensure they are free from vulnerabilities.
5. **Nikto**:
   - An open-source web server scanner.
   - Features: Scans for potentially dangerous files/programs, outdated versions, and server configuration issues.
   - Example Use: Quickly scanning web servers for known vulnerabilities.

Muhammad Tayab's 3rd week of 2nd day report

Example Scenario

Using OWASP ZAP to Scan a Web Application:

1. **Setup and Install OWASP ZAP**:
   - Download and install OWASP ZAP from the official website.
2. **Start OWASP ZAP**:
   - Launch the OWASP ZAP application and configure the proxy settings to intercept browser traffic.
3. **Spidering the Application**:
   - Use the spidering feature to crawl the web application and discover all the pages and inputs.
4. **Active Scanning**:
   - Run the active scanner to test all the discovered endpoints for vulnerabilities.
5. **Analyze Results**:
   - Review the scan results, which will list identified vulnerabilities like SQL injection, XSS, etc.
6. **Remediation**:
   - Address the identified vulnerabilities by updating the code, configurations, or applying patches.

Techniques Summary:

1. **Static Analysis (SAST)**: Analyze source code for vulnerabilities.
2. **Dynamic Analysis (DAST)**: Test running applications for security issues.
3. **Interactive Analysis (IAST)**: Real-time testing during application execution.
4. **Composition Analysis (SCA)**: Check third-party components for known vulnerabilities.
5. **Penetration Testing**: Manual testing to simulate attacks.
6. **Fuzz Testing**: Random data input to find unknown vulnerabilities.

Burp Suite

Burp Suite is a comprehensive and popular platform for web application security testing. It provides a range of tools for identifying and exploiting vulnerabilities in web applications. Developed by PortSwigger, Burp Suite is widely used by security professionals for both manual and automated testing.

Key Features:

1. **Proxy**: Intercepts and modifies HTTP/S traffic between the browser and web servers.
2. **Spider**: Automatically crawls web applications to discover content and functionality.
3. **Scanner**: Automated scanner for detecting a wide range of vulnerabilities, including SQL injection, cross-site scripting (XSS), and more.
4. **Intruder**: A powerful tool for performing automated customized attacks on web applications.
5. **Repeater**: Allows manual manipulation and resending of individual HTTP requests.
6. **Sequencer**: Analyzes the randomness of tokens and session IDs.

7. **Extender**: Integrates with third-party extensions to extend Burp Suite's capabilities.
8. **Collaborator**: Helps detect issues like server-side request forgery (SSRF) and out-of-band vulnerabilities.

Example Use Case:

1. **Intercepting and Modifying Traffic**:
   - Configure the browser to use Burp Suite as a proxy.
   - Intercept HTTP/S requests and responses to examine and modify them.
2. **Scanning for Vulnerabilities**:
   - Use the automated scanner to identify common vulnerabilities like XSS, SQL injection, etc.
3. **Manual Testing**:
   - Use the Intruder and Repeater tools to manually test and exploit vulnerabilities.

Acunetix

Acunetix is a commercial web vulnerability scanner designed to identify security issues in web applications and websites. It provides both automated and manual testing tools and integrates well into the CI/CD pipeline, making it suitable for continuous security assessment.

Key Features:

1. **Automated Scanning**: Comprehensive scanning for over 7,000 vulnerabilities, including SQL injection, XSS, and more.
2. **DeepScan**: Crawls and analyzes single-page applications (SPAs) and sites that heavily use JavaScript.
3. **Login Sequence Recorder**: Allows the scanner to log into applications and perform authenticated scans.
4. **Vulnerability Management**: Provides detailed reports and remediation advice for identified vulnerabilities.
5. **Integration**: Integrates with various CI/CD tools, issue trackers, and other security tools.
6. **Proof-of-Concept**: Generates proof-of-concept (PoC) exploits for detected vulnerabilities to demonstrate their impact.
7. **Network Security Scanning**: In addition to web applications, Acunetix can also perform network security scans.

Example Use Case:

1. **Initial Setup**:
   - Configure Acunetix to scan the target web application.
   - Define scan parameters, including authentication if required.
2. **Automated Scanning**:
   - Initiate an automated scan to identify vulnerabilities in the web application.
   - Review the scan results and PoC exploits provided by Acunetix.
3. **Integration with CI/CD**:

   - Integrate Acunetix with Jenkins, GitLab CI, or other CI/CD tools to ensure continuous security testing.
4. **Remediation and Re-Scanning**:
   - Use the detailed reports and remediation advice to fix the identified vulnerabilities.
   - Re-scan the application to verify that the issues have been resolved.

Comparison and Use Cases:

- **Burp Suite** is ideal for hands-on penetration testing, allowing detailed manual testing and exploitation of web application vulnerabilities.
- **Acunetix** excels in automated scanning and integration with development pipelines, making it suitable for continuous security assessment in agile environments.

Both tools are essential for comprehensive web application security testing, and they can be used together to combine the strengths of automated and manual testing methodologies.

SQL Injection

SQL Injection is a type of security vulnerability that allows an attacker to interfere with the queries that an application makes to its database. It typically occurs when user input is not properly sanitized before being included in SQL queries. This can result in unauthorized access to database information, data manipulation, or even the execution of administrative operations on the database.

How It Works:

- **Direct Injection**: Directly inserting malicious SQL code into input fields, such as login forms.
- **Union-based Injection**: Using the `UNION` SQL operator to combine the results of two or more SELECT statements.
- **Error-based Injection**: Forcing the database to generate error messages that provide information about the database structure.
- **Blind Injection**: When the application does not display database errors, attackers infer results based on the application's response behavior.

Example:

If an application executes the following SQL query using user input:

```
SELECT * FROM users WHERE username = '$username' AND password = '$password';
```

An attacker could enter `admin' --` as the username, leading to:

```
SELECT * FROM users WHERE username = 'admin' --' AND password = '';
```

The `--` comments out the rest of the query, potentially granting access without a password.

SQLMap

SQLMap is an open-source penetration testing tool that automates the process of detecting and exploiting SQL injection vulnerabilities. It supports a wide range of databases and SQL injection techniques.

Key Features:

- **Automated Detection and Exploitation**: SQLMap can automatically detect and exploit SQL injection vulnerabilities.
- **Database Fingerprinting**: Identifies the type and version of the database.
- **Data Extraction**: Extracts data from the database using different techniques.
- **Access and Control**: Reads and writes files on the database server, executes commands on the server, etc.
- **Integration**: Can be integrated into other tools and scripts for automated testing.

Example Usage:

To test a URL for SQL injection:

```
sqlmap -u "http://example.com/vulnerable_page.php?id=1" --batch
```

This command scans the specified URL for SQL injection vulnerabilities.

Cross-Site Scripting (XSS)

Cross-Site Scripting (XSS) is a type of security vulnerability typically found in web applications. It allows attackers to inject malicious scripts into web pages viewed by other users. These scripts can steal cookies, session tokens, or other sensitive information, and can also perform actions on behalf of the users without their consent.

Types of XSS:

- **Stored XSS**: The malicious script is permanently stored on the target server (e.g., in a database) and executed when a user views the stored data.
- **Reflected XSS**: The malicious script is reflected off a web server, such as in an error message, search result, or any other response that includes user input.
- **DOM-based XSS**: The vulnerability exists in client-side code rather than server-side code.

Example:

A comment section that does not sanitize input properly:

```
<input type="text" name="comment">
```

An attacker could input:

```
<script>alert('XSS');</script>
```

When another user views the comment, the script is executed.

XSS Detection

Manual Testing:

- **Input Fields**: Test input fields by injecting payloads like `<script>alert('XSS');</script>`.
- **Response Analysis**: Observe the application's response to see if the script executes.

Automated Tools:

- **Burp Suite**: Its scanner can detect XSS vulnerabilities during automated scans.
- **OWASP ZAP**: An open-source tool for finding vulnerabilities in web applications, including XSS.
- **XSSer**: An automated framework to detect, exploit, and report XSS vulnerabilities.

Example Using Burp Suite:

1. **Proxy Configuration**: Set up Burp Suite as a proxy and browse the target application.
2. **Scan**: Use the scanner to automatically test for XSS vulnerabilities.
3. **Intruder**: Use the Intruder tool to manually test specific input points with various XSS payloads.

Summary

- **SQL Injection**: Exploits vulnerabilities in SQL queries to manipulate databases.
- **SQLMap**: Automates the detection and exploitation of SQL injection vulnerabilities.
- **XSS (Cross-Site Scripting)**: Injects malicious scripts into web pages, affecting other users.
- **Detection**: Can be done manually or with automated tools like Burp Suite, OWASP ZAP, and XSSer.

Threat Intelligence Integration

Threat Intelligence Integration involves incorporating data and insights about current and emerging threats into an organization's security processes. This integration helps organizations understand potential risks, prepare for attacks, and respond effectively. It encompasses gathering, analyzing, and applying information about threats from various sources to enhance the organization's security posture.

Benefits:

- **Proactive Defense**: Identifies threats before they impact the organization.
- **Enhanced Incident Response**: Improves the ability to detect and respond to incidents quickly.
- **Contextual Awareness**: Provides context about threats, helping to prioritize and address the most significant risks.
- **Improved Decision-Making**: Informs strategic and tactical decisions regarding cybersecurity investments and policies.

Muhammad Tayab's 3rd week of 2nd day report

Tools and Techniques for Threat Intelligence Integration

1. Threat Intelligence Platforms (TIPs)

- **Description**: TIPs collect, aggregate, and analyze threat data from multiple sources.
- **Examples**:
    - **ThreatConnect**: Provides threat intelligence management, automation, and analytics.
    - **Recorded Future**: Uses machine learning to analyze and visualize threat data.

2. Security Information and Event Management (SIEM) Systems

- **Description**: SIEMs collect and analyze log data from various sources to detect and respond to security incidents.
- **Examples**:
    - **Splunk**: Offers real-time monitoring, log management, and threat detection.
    - **IBM QRadar**: Provides log analysis, threat intelligence, and incident response.

3. Endpoint Detection and Response (EDR) Tools

- **Description**: EDR tools monitor endpoints for suspicious activities and provide threat detection and response capabilities.
- **Examples**:
    - **CrowdStrike Falcon**: Detects and responds to threats on endpoints.
    - **Carbon Black**: Offers endpoint protection, threat hunting, and incident response.

4. Network Traffic Analysis (NTA) Tools

- **Description**: NTA tools analyze network traffic to detect anomalies and potential threats.
- **Examples**:
    - **Darktrace**: Uses machine learning to detect threats based on network behavior.
    - **Vectra AI**: Provides real-time detection of cyberattacks through AI-driven network analysis.

5. Threat Intelligence Feeds

- **Description**: Data streams that provide information about current threats, including IP addresses, domains, URLs, and malware hashes.
- **Examples**:
    - **AlienVault OTX**: A free threat intelligence sharing platform.
    - **IBM X-Force Exchange**: Provides threat intelligence feeds and reports.

6. Open Source Intelligence (OSINT) Tools

- **Description**: Tools that gather publicly available information from the internet to identify potential threats.
- **Examples**:
    - **Maltego**: Visual link analysis tool for OSINT and threat intelligence.

- **SpiderFoot**: Automates the collection of OSINT data.

Techniques for Threat Intelligence Integration

1. **Automated Threat Data Ingestion**
   - Integrate threat intelligence feeds into security tools to automate the ingestion of threat data.
   - Example: Configuring a SIEM to ingest data from threat intelligence feeds for real-time analysis.
2. **Correlation and Analysis**
   - Use SIEM and TIPs to correlate threat data with internal logs and events to identify potential threats.
   - Example: Correlating threat intelligence data with firewall logs to detect malicious IP addresses.
3. **Threat Hunting**
   - Proactively search for threats within an organization's network using EDR tools and threat intelligence.
   - Example: Using CrowdStrike Falcon to hunt for indicators of compromise (IoCs) provided by threat intelligence.
4. **Incident Response**
   - Use threat intelligence to inform and enhance incident response efforts.
   - Example: Using Recorded Future to provide context and additional details during a security incident investigation.
5. **Sharing and Collaboration**
   - Share threat intelligence with other organizations and industry groups to enhance collective defense.
   - Example: Participating in threat intelligence sharing platforms like AlienVault OTX.

Examples of Threat Intelligence Integration

Example 1: Financial Sector

A bank integrates threat intelligence into its SIEM (Splunk) to detect and respond to phishing attacks. The SIEM correlates threat intelligence data with email logs to identify and block malicious emails before they reach employees.

Example 2: Healthcare Sector

A hospital uses CrowdStrike Falcon for endpoint detection and response. Threat intelligence feeds from IBM X-Force Exchange are integrated into Falcon to enhance detection of ransomware attacks targeting hospital systems.

Example 3: Manufacturing Sector

A manufacturing company uses Darktrace for network traffic analysis. Darktrace integrates with Recorded Future to identify and respond to advanced persistent threats (APTs) targeting industrial control systems.

By integrating threat intelligence, organizations can stay ahead of cyber threats and ensure a robust security posture.

Wireless Network Scans

Wireless network scanning is the process of detecting and analyzing wireless networks and their components. This involves identifying available networks, their signal strength, security settings, and other characteristics. Wireless network scans are used by network administrators for managing and securing networks and by ethical hackers for identifying potential vulnerabilities.

Key Objectives of Wireless Network Scanning:

1. **Discovering Wireless Networks**: Identifying the SSIDs (Service Set Identifiers) of nearby networks.
2. **Assessing Signal Strength**: Measuring the strength of wireless signals to understand coverage and performance.
3. **Determining Security Settings**: Checking the security protocols in use (e.g., WEP, WPA, WPA2).
4. **Identifying Rogue Access Points**: Detecting unauthorized access points within the network.
5. **Channel Analysis**: Understanding which channels are in use to avoid interference.

Wi-Fi Technologies

1. 802.11 Standards

- **802.11a**: Operates at 5 GHz, up to 54 Mbps.
- **802.11b**: Operates at 2.4 GHz, up to 11 Mbps.
- **802.11g**: Operates at 2.4 GHz, up to 54 Mbps.
- **802.11n**: Operates at both 2.4 GHz and 5 GHz, up to 600 Mbps.
- **802.11ac**: Operates at 5 GHz, up to several Gbps.
- **802.11ax (Wi-Fi 6)**: Operates at both 2.4 GHz and 5 GHz, offers higher efficiency and throughput.

2. Wi-Fi Security Protocols

- **WEP (Wired Equivalent Privacy)**: An outdated security protocol that is vulnerable to many types of attacks.
- **WPA (Wi-Fi Protected Access)**: Introduced as an interim solution to address the weaknesses of WEP.
- **WPA2 (Wi-Fi Protected Access II)**: Provides stronger security through the use of AES encryption and CCMP.
- **WPA3 (Wi-Fi Protected Access III)**: The latest security standard, offering enhanced protections over WPA2.

Muhammad Tayab's 3rd week of 2nd day report

Security Technologies and Terms

1. CCMP (Counter Mode Cipher Block Chaining Message Authentication Code Protocol)

- **Description**: An encryption protocol used in WPA2 for data confidentiality, integrity, and authentication.
- **Function**: Uses AES in counter mode for encryption and cipher block chaining message authentication for integrity.

2. AES (Advanced Encryption Standard)

- **Description**: A symmetric encryption algorithm widely used across the globe.
- **Function**: Provides strong encryption with key sizes of 128, 192, or 256 bits.

3. WPA (Wi-Fi Protected Access)

- **Description**: A security protocol designed to improve upon the security features of WEP.
- **Function**: Uses TKIP (Temporal Key Integrity Protocol) for encryption.

4. WPA2 (Wi-Fi Protected Access II)

- **Description**: An enhanced version of WPA that requires the use of stronger wireless encryption protocols (AES) and introduces CCMP.
- **Function**: Provides robust security for wireless networks by enforcing stronger encryption and integrity protocols.

5. WPA3 (Wi-Fi Protected Access III)

- **Description**: The latest Wi-Fi security standard that enhances WPA2's protections and introduces new features.
- **Function**:
    - **Enhanced Protection for Open Networks**: Through Opportunistic Wireless Encryption (OWE).
    - **Stronger Encryption Algorithms**: Uses 192-bit security suite for enterprise environments.
    - **Protection Against Brute-Force Attacks**: Through a feature called SAE (Simultaneous Authentication of Equals).

Summary of Wi-Fi Security Protocols

Protocol	Encryption	Authentication	Vulnerabilities	Notes
WEP	RC4	Shared Key	Weak encryption, easily broken	Deprecated, not secure
WPA	TKIP/RC4	PSK/802.1X	Susceptible to some attacks	Interim solution before WPA2

Muhammad Tayab's 3rd week of 2nd day report

Protocol	Encryption	Authentication	Vulnerabilities	Notes
WPA2	AES/CCMP	PSK/802.1X	Stronger than WPA	Current standard for most networks
WPA3	AES-GCMP	SAE/192-bit	Strongest protection	Latest standard

Wireless Network Scanning Tools

1. **Aircrack-ng**: A suite of tools for auditing wireless networks, including monitoring, attacking, testing, and cracking.
2. **Kismet**: A wireless network detector, sniffer, and intrusion detection system.
3. **Wireshark**: A network protocol analyzer that can capture and analyze wireless traffic.
4. **NetStumbler**: A tool for Windows that detects 802.11a/b/g WLANs.
5. **Acrylic Wi-Fi**: A Wi-Fi analyzer for Windows that provides detailed information about wireless networks.

Examples of Wireless Network Scanning

Using Aircrack-ng on Kali Linux

```
# Put the wireless interface into monitor mode
airmon-ng start wlan0

# Scan for wireless networks
airodump-ng wlan0mon

# Capture traffic from a specific network
airodump-ng -c <channel> --bssid <AP_MAC> -w <output_file> wlan0mon

# Deauthenticate a client to capture the handshake
aireplay-ng --deauth 10 -a <AP_MAC> -c <Client_MAC> wlan0mon

# Crack the WPA/WPA2 handshake
aircrack-ng -w <wordlist> -b <AP_MAC> <output_file>.cap
```

Wireless network scanning is essential for both network administrators to maintain and secure networks and for ethical hackers to identify and mitigate vulnerabilities. The use of strong encryption protocols like WPA3 and AES ensures that wireless communications remain secure against most common attacks.