RV College of Engineering®

Go, change the world

CYX55TBD-Introduction to Vulnerability Assessment & Penetration Testing

Unit 5: Chapter- 28 Collecting Malware and Initial Analysis

Course Incharge: Dr.Mohana
Department of Computer Science & Engineering (Cyber Security)
RV College of Engineering, Bangalore-560059

Unit - V Syllabus

Unit –V	07 Hrs
Client-Side Browser Exploits: Why client-side vulnerabilities are interesting, Internet explorer security concepts, history of client- side exploits and latest trends, finding new browser-based vulnerabilities heap spray to exploit, protecting your self from clients side exploit. Malware Analysis: Collecting Malware and Initial Analysis: Malware, Latest Trends in Honeynet Technology, Catching Malware: Setting the Trap, Initial Analysis of Malware.	

Learning Objectives

- Malware
- Latest trends in honeynet technology
- Catching malware: setting the trap
- Initial analysis of malware

Malware

- Unintended and unsolicited installation of software on a system without the user knowing or wanting it.
- Understanding, examining, and dissecting malicious software to identify its behavior, origin, and impact.
- To detect and defend against malware effectively.

Types of Malware

Virus:

- Malicious software (malware) designed to spread from one device to another.
- infect program and perform some unwanted function.
- Some are easy to detect and remove from a system.
- whereas others are very difficult to detect and remove.
- Some viruses use polymorphic (changing) technology to morph as they move from system to system, thereby prolonging their detection.

Characteristics:

- **Replication**: A virus can replicate itself by attaching to other programs or files.
- **Activation**: It requires execution by the user (e.g., opening an infected file) to activate.
- **Payload**: The destructive part of the virus, which might delete files, alter data, or perform other malicious activities.
- **Infection Mechanism**: Viruses spread through infected files, emails, removable drives, or compromised websites.

Trojan Horse

- Type of malicious software that disguises itself as a legitimate or harmless program to deceive users into installing or executing it.
- Trojan does not self-replicate. Instead, it relies on the victim to activate it, often through social engineering tactics.

Characteristics

- **Deceptive Appearance**: It appears to be a useful or benign file or program (e.g., a game, software update, or email attachment).
- **Hidden Payload**: Once executed, it performs malicious activities such as stealing data, installing additional malware, or allowing unauthorized access to the system.
- **No Self-Replication**: Unlike viruses or worms, a Trojan does not spread on its own.

Common Types of Trojans:

- **Backdoor Trojans**: Open a backdoor to the system, allowing attackers to gain remote control.
- **Banking Trojans**: Designed to steal financial information, such as credit card details and online banking credentials.
- **Downloader Trojans**: Download and install other malicious software onto the infected device.
- **Spyware Trojans**: Monitor user activity, such as keystrokes (keylogging) or screen captures.
- **Ransom Trojans**: Encrypt files and demand payment (ransomware) to restore access.
- **Rootkit Trojans**: Hide other malware or processes to evade detection.

Trojans Spread:

- **Email Attachments**: Malicious attachments pretending to be documents or software.
- **Infected Websites**: Clicking on fake ads or downloading files from untrusted sites.
- **Pirated Software**: Many Trojans are embedded in cracked software or illegal downloads.
- **Social Media Links**: Links in messages or posts that lead to Trojan downloads.

Signs of a Trojan Infection:

- Slow computer performance.
- Unexpected pop-ups or ads.
- New programs or icons you didn’t install.
- Unauthorized changes to system settings.
- Unexplained network activity.

worms

- Type of malware that can self-replicate and spread across systems without requiring a host program or user intervention.
- Unlike viruses, which need a user to execute a file or program to spread, worms exploit network vulnerabilities, operating systems, or software flaws to propagate automatically.

Characteristics:

- **Self-Replication:** Worms duplicate themselves and spread rapidly.
- **No Host Required:** They are standalone programs that don’t need to attach to other files or programs.
- **Exploits Vulnerabilities:** Often use network security flaws, such as weak passwords or outdated software, to infiltrate systems.
- **Payload Delivery:** Some worms carry destructive payloads that delete files, install backdoors, or overload systems. Others simply spread and consume resources, causing disruptions.

Worms Working:

Entry Point: Worms typically enter a system via email attachments, malicious links, or infected software.

Propagation: Once inside, they exploit vulnerabilities in the network or operating system to spread to other devices.

Execution: They execute malicious activities such as:

Consuming system resources (disk space, CPU, bandwidth).

Installing backdoors for further attacks.

Delivering additional malware, like ransomware.

Signs of a Worm Infection:

- Decreased network performance (slow internet or file transfers).
- Unexplained system crashes or freezes.
- High CPU or memory usage.
- Unexpected files, processes, or network connections.

Examples of Famous Worm Attacks:

- **Morris Worm (1988)**: One of the first worms on the internet, it caused major disruptions by spreading rapidly.
- **ILOVEYOU (2000)**: Spread through email with the subject "ILOVEYOU," causing billions in damages worldwide.
- **Code Red (2001)**: Exploited vulnerabilities in Microsoft IIS web servers, defaced websites, and launched DDoS attacks.
- **Stuxnet (2010)**: A highly sophisticated worm that targeted industrial systems, particularly Iran’s nuclear facilities.
- **WannaCry (2017)**: Combined a worm with ransomware, exploiting SMB vulnerabilities to infect systems globally.

Spyware

- Spyware is malicious software designed to secretly collect information about a user, their device, or their online activities without their knowledge or consent.

*Characteristics***:**

- **Stealthy Operation**: Runs silently in the background to avoid detection.
- **Data Collection**: Gathers personal information such as:
  - Login credentials
  - Credit card numbers
  - Browsing habits
  - Sensitive documents
- **Transmission**: Sends collected data to a **third party for malicious purposes** like identity theft or targeted advertising.

Types of Spyware:

- **Keyloggers**: Record keystrokes to capture passwords, credit card numbers, and more.
- **Tracking Cookies**: Monitor browsing habits and collect data for advertisers.
- **System Monitors**: Record user activity, including screenshots, app usage, and file access.
- **Banking Trojans**: Specifically designed to steal financial information.

Adware

- Adware is software designed to deliver unwanted advertisements, often in the form of pop-ups or banners. It may also collect user data to tailor ads.

Characteristics:

- **Aggressive Advertising**: Bombards users with intrusive ads.
- **Performance Impact**: Slows down systems by consuming resources.
- **Data Collection**: Tracks browsing habits to deliver personalized ads.

Is Adware Always Malicious?

Is Adware Always Malicious?

- Not always. Some adware is legitimate and included in free software as a way to fund development.
- However, when installed without user consent or combined with spyware, it becomes malicious.

Spyware and Adware Spread:

- **Free Software**: Bundled with "free" applications or downloads.
- **Malicious Websites**: Clicking on unsafe links or pop-ups.
- **Email Attachments**: Opening infected attachments.
- **Fake Updates**: Installing updates from untrusted sources.
- **Drive-By Downloads**: Downloaded automatically when visiting compromised websites.

Signs of Infection:

For Spyware:

- Slow system performance.
- Unusual network activity.
- Unauthorized changes to settings or accounts.

For Adware:

- Frequent pop-up ads, even when offline.
- Unexpected changes to browser settings (e.g., homepage or search engine).
- Installation of unknown programs or toolbars.

Malware Analysis

- Malware analysis involves understanding, examining, and dissecting malicious software to identify its behavior, origin, and impact.
- It helps cybersecurity professionals detect and defend against malware effectively.

Key Tools for Malware Analysis

- **Static Tools**: Strings, PE Explorer, Binwalk.
- **Dynamic Tools**: Cuckoo Sandbox, ProcMon, SysInternals.
- **Network Tools**: Wireshark, Fiddler, tcpdump.
- **Reverse-Engineering**: IDA Pro, Ghidra, Radare2.

Types of Malware Analysis

Static Analysis

Examines malware binaries without executing the code.

Tools: strings, IDA Pro, Ghidra, objdump.

Dynamic Analysis

Involves running the malware in a controlled environment (sandbox) to observe its behavior.

Tools: Cuckoo Sandbox, VMware, Process Monitor.

Behavioral Analysis

Focuses on understanding how malware interacts with the system (network traffic, file system, registry).

Tools: Wireshark, Regshot.

Code Analysis

Reverse-engineering the malware code to identify functions and logic.

Tools: Ghidra, IDA Pro.

Steps for Malware Analysis

- **Set Up an Isolated Lab Environment**
    - Use virtual machines (VMware, VirtualBox) and sandboxes.
    - Avoid direct execution on production systems.
- **Perform Static Analysis**
    - Extract strings, hashes, and file metadata.
    - Use tools like VirusTotal for file reputation.
- **Dynamic Analysis**
    - Run malware in a sandbox.
    - Monitor its actions on network, files, and system processes.
- **Behavioral and Code Analysis**
    - Study malware's persistence mechanisms, obfuscation, and encryption techniques.
    - Analyze assembly code for logic.

Common Malware Techniques

- **Code Obfuscation**: Making code harder to read (e.g., encrypted payloads).
- **Polymorphism/Metamorphism**: Changing structure to evade detection.
- **Fileless Malware**: Resides in memory to avoid traditional detection.
- **Rootkits**: Hides malicious processes/files at the OS level.

Collecting Malware

- Critical for understanding the nature and impact of a malicious program.
- Proper handling and safe practices are essential to avoid infection or spreading malware.

Sources of Malware Samples:

Public Repositories:

VirusTotal: Online file-scanning and analysis tool that allows access to malware hashes.

MalwareBazaar: Community-driven repository for malware samples.

Hybrid Analysis: Provides automated analysis of malware samples.

Any.Run: Interactive malware analysis sandbox.

TheZoo: Open-source repository for live malware samples.

Honeypots:

Deploy honeypots to attract and capture malware in controlled environments.

Tools: **Dionaea**, **Kippo**, or **Cuckoo Sandbox**.

Email Attachments:

Analyze suspicious email attachments with caution (phishing campaigns).

Network Traffic:

Use network packet capture tools (e.g., Wireshark) to capture malicious payloads delivered through the network.

Safe Handling of Malware Samples

- **Isolate Environment**:
  - Use virtual machines (VMware, VirtualBox) or sandboxes.
  - Disable networking or use NAT with strict controls.
  - Use snapshots to revert to clean states.
- **Use Dedicated Tools**:
  - Store malware samples in password-protected archives (.zip with password infected).
  - Use tools like 7-Zip or WinRAR for this.
- **Malware Storage**:
  - Maintain a secure, offline malware repository.
  - Use labels or hashes to identify files.
- **Do Not Open on Production Machines**:
  - Ensure malware never interacts with operational systems.

Initial Malware Analysis

- Gathering basic information without deep reverse engineering

Static Analysis (Without Execution)

- Examines the malware file without running it.

Tools: strings, PE Explorer, Binwalk, Ghidra, objdump.

Dynamic Analysis (Executing Malware in Controlled Environments)

- Running malware in a sandbox to observe its behavior.

Tools: Cuckoo Sandbox, Any.Run, Process Monitor, Wireshark.

Steps in Static Analysis

1. **Hashing**
   - Compute cryptographic hashes (MD5, SHA256) to identify known malware.
   - **Tool:** HashCalc, md5sum.
   - Check against malware databases like VirusTotal or MalwareBazaar.
2. **File Type and Metadata**
   - Identify the file format and metadata.
   - **Tools:** file, ExifTool.
3. **Extract Strings**
   - Look for readable strings that may indicate URLs, IPs, commands, or obfuscation.
   - **Tool:** strings (Linux) or Floss for obfuscated strings.
4. **Disassemble Code**
   - Analyze the malware's assembly code for functions or behaviors.
   - **Tools:** IDA Pro, Ghidra, Radare2.
5. **Analyze Import Table**
   - Check for imported functions (e.g., CreateProcess, WriteFile) that suggest malicious behaviors.
   - **Tool:** PEStudio.

Steps in Dynamic Analysis

1. **Behavioral Observation**
   - Monitor file creation, network communication, registry modifications, and process activity.
2. **Capture Network Activity**
   - Identify connections to malicious domains or IPs.
   - **Tool:** Wireshark, tcpdump.
3. **Monitor Processes and File System**
   - Observe process creation, memory usage, and file system changes.
   - **Tools:** ProcMon, Process Explorer.
4. **Check Persistence Mechanisms**
   - Investigate for registry keys, scheduled tasks, or services created for persistence.

Tools for Initial Malware Analysis

Category	Tool Name	Purpose
Static Analysis	`strings`, `PEStudio`	Extract metadata, strings, imports.
Dynamic Analysis	`Cuckoo Sandbox`	Run malware in isolation.
Network Traffic	`Wireshark`	Monitor network activity.
Process Monitoring	`ProcMon`, `Process Explorer`	Monitor processes and I/O.
Hashing and Signatures	`HashCalc`, `VirusTotal`	Check hashes and scan for detection.
Reverse Engineering	`IDA Pro`, `Ghidra`	Disassemble and inspect malware.

After Initial Analysis

- **Classify Malware**: Identify its type (e.g., ransomware, trojan, worm).
- **Document Findings**: Record behaviors, hashes, file paths, and network IOCs (Indicators of Compromise).
- **Report and Share**: Share IOCs with threat intelligence platforms (e.g., MISP, VirusTotal).

Malware Defensive Techniques

- Used to **protect systems, networks, and data** from malicious software.

Types

1. **Proactive Techniques -** Preventing malware infections before they occur
2. **Reactive Techniques -** Mitigating and responding to malware infections
3. **Advanced Techniques -** cutting-edge technology and approaches
4. **User Awareness and Training**
5. **Legal and Policy Measures**

1. Proactive Techniques

Endpoint Protection

- **Antivirus and Anti-Malware Software**: Install and regularly update antivirus tools to detect and block malware.
- **Host-based Intrusion Prevention Systems (HIPS)**: Monitor and prevent suspicious activities on endpoints.
- **Application Whitelisting**: Restrict systems to run only approved software.

Network Security

- **Firewall Configuration**: Filter incoming and outgoing traffic to block malicious data.
- **Network Segmentation**: Limit the spread of malware by isolating sensitive systems.
- **Intrusion Detection and Prevention Systems (IDPS)**: Identify and respond to unusual patterns in network traffic.

Software and System Hardening

- **Patch Management**: Regularly update operating systems and applications to close vulnerabilities.
- **Least Privilege Access**: Limit user access rights to reduce the attack surface.
- **Secure Configuration**: Disable unnecessary services and enforce secure settings.

Threat Intelligence

- **Reputation Services**: Use databases to block known malicious IPs, domains, and files.
- **Honeypots**: Deploy decoy systems to attract and analyze malware.

2. Reactive Techniques

Incident Response

- **Detection and Containment**: Quickly identify malware infections and isolate affected systems.
- **Eradication**: Remove malware using specialized tools or manual methods.
- **Recovery**: Restore data and systems from clean backups.

Malware Analysis

- **Dynamic Analysis**: Execute malware in a controlled environment (sandbox) to understand its behavior.
- **Static Analysis**: Analyze malware binaries without executing them to identify characteristics and potential threats.

Backup and Restore

- **Regular Backups**: Maintain frequent and secure backups of critical data to minimize the impact of ransomware and other destructive malware.
- **Immutable Backups**: Use backups that cannot be altered to ensure integrity.

Behavioral Monitoring

- **Anomaly Detection**: Use machine learning and behavioral analytics to identify unusual activities that could signal a malware attack.

3. Advanced Techniques

Endpoint Detection and Response (EDR)

- Continuously monitor and analyze endpoint activities for signs of advanced threats.

Threat Hunting

- Proactively search for indicators of compromise (IoCs) within networks to detect and neutralize threats early.

Zero Trust Architecture

- Apply strict access controls and assume all traffic is potentially malicious.

Artificial Intelligence (AI) and Machine Learning

- Employ AI models to detect previously unseen malware variants through behavioral patterns and heuristics.

Virtualization and Sandboxing

- Execute files in isolated environments to determine if they are malicious before deployment.

4. User Awareness and Training

- Educating users on identifying phishing attempts, avoiding suspicious downloads, and practicing good cybersecurity hygiene is a cornerstone of malware defense.

5. Legal and Policy Measures

- Establish organizational policies for acceptable use, incident handling, and regular audits.
- Stay updated with legal frameworks and compliance requirements related to malware defense.

Top 25 CWEs Truly the Most Dangerous Software Weaknesses in 2024?

https://vulncheck.com/blog/cwe-top-25-2024

- Rootkits
- Packers
- Protective Wrappers with Encryption
- VM Detection

Rootkits

- Category of software that hides itself and other software from system administrators in order to perform some nefarious task.
- stealthy malware designed to gain unauthorized access and hide their presence on a system, often by manipulating the operating system.

Defensive Techniques for Rootkits:

- **Kernel Integrity Monitoring**: Use tools that verify the integrity of critical system files and kernel modules.
- **Behavioral Analysis**: Monitor suspicious activities like unauthorized privilege escalation or abnormal process injections.
- **Memory Scanning**: Perform in-depth scans of system memory to detect hidden malicious processes or drivers.
- **Rootkit Removal Tools**: Use specialized tools like GMER or Microsoft's RootkitRevealer to detect and remove rootkits.
- **Reinstallation of OS**: For severe infections, reinstalling the operating system is often the only way to ensure complete removal.

Packers

- Packers are used to “pack” or compress the Windows PE file format.
- Packers compress or encrypt malware payloads to obfuscate their content and evade static analysis by antivirus software.
- Most common packers are UPX, ASPack, tElock

Defensive Techniques for Packers:

- **Unpacking Tools**: Use tools such as UPX, OllyDbg, or IDA Pro to extract the original code from packed files.
- **Dynamic Analysis**: Execute the packed file in a sandbox environment to observe its behavior and determine its intent.
- **Heuristic Analysis**: Employ heuristic-based antivirus engines that can detect patterns indicative of packed malware.
- **Entropy Analysis**: Measure the entropy (randomness) of files to flag unusually high values typical of packed files.

Protective Wrappers with Encryption

- Malware uses encryption to protect its payload, often wrapping malicious code in layers of encryption to prevent analysis.
- Some hackers use tools such as the following to wrap their binary with encryption: Burneye, Shiva

Defensive Techniques for Protective Wrappers:

- **Decryption Tools**: Use cryptographic analysis tools to strip away layers of encryption and reveal the payload.
- **Memory Dumping**: Analyze the malware when it's running in memory, as the decrypted payload often resides in RAM during execution.
- **Threat Intelligence Sharing**: Collaborate with other security teams to share decryption keys or known wrapper techniques.
- **Behavior-Based Detection**: Focus on the actions the malware performs rather than its static signature.

VM Detection

- Malware often includes mechanisms to detect if it is running in a virtual machine (VM) or sandbox.
- If a VM is detected, the malware may alter its behavior or remain dormant to evade analysis.

Defensive Techniques for VM Detection:

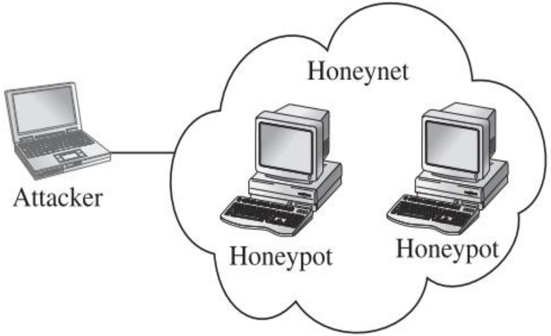
- **Anti-VM Evasion Tools**: Modify VM configurations to mimic physical machines and bypass basic VM detection checks.
- **Hardware-Assisted Virtualization**: Use advanced virtualization platforms that can better mask the presence of a VM.
- **Stealthy Sandboxing**: Design sandboxes that simulate real-world environments to deceive malware into executing normally.
- **Binary Instrumentation**: Intercept and analyze malware's system calls and behaviors without relying solely on virtualization.
- **Dynamic Renaming of VM Artifacts**: Remove or rename common VM-related artifacts such as "VMware" or "VirtualBox" to trick malware.

Unified Approaches to Counter These Techniques:

- **AI and Machine Learning**: Analyze large datasets to identify subtle patterns indicative of rootkits, encrypted malware, or VM-aware malware.
- **Threat Intelligence Platforms**: Leverage global threat databases to keep up-to-date on new evasion tactics.
- **Code Emulation**: Emulate suspicious code in controlled environments to understand its functionality without executing it directly.
- **Layered Security**: Combine multiple defensive layers, such as network, endpoint, and application-level protections, to create a comprehensive defense system.

Honeypots

- A honeypot is a decoy system or network resource designed to attract, detect, and analyze cyber threats.
- It mimics legitimate systems, enticing attackers to interact with it while collecting information about their techniques, tools, and intentions.
- Honeypots can vary in complexity and purpose
- Honeypots are decoy systems placed in the network for the sole purpose of attracting hackers.
- The systems are not valuable and contain no sensitive information, but they look like they are valuable.
- “honeypots” because once the hackers put their hands in the pot and taste the honey, they keep coming back for more.



Cat and mouse game:

- battle between defenders (security experts) and attackers (hackers or malware creators).
- This cycle typically involves innovation, countermeasures, and strategic maneuvering by both sides

- **Attackers**: Continuously develop new techniques like malware, ransomware, or phishing strategies to bypass security measures.
- **Defenders**: Respond with updated firewalls, antivirus software, and threat detection systems, forcing attackers to adapt again.

Types of Honeypots

1. Based on Interaction Level

Low-Interaction Honeypots:

- Simulate limited functionalities of systems or services.
- Easy to set up and maintain.
- **Examples:** Simulated SSH or FTP servers.
- **Use:** Detecting automated attacks or simple reconnaissance.

High-Interaction Honeypots:

- Offer a realistic environment, such as a fully functional operating system.
- Allow attackers to interact deeply, providing richer intelligence.
- Require more resources and risk management.
- **Use:** Studying advanced threat actors and sophisticated malware.

Medium-Interaction Honeypots:

- Strike a balance between low and high interaction.
- Provide more interactivity than low-level honeypots but without the complexity of high-level setups.

2. Based on Deployment Goals

Research Honeypots:

- Used to study attacker behaviors, malware, and exploit trends.
- Typically deployed in controlled environments by cybersecurity researchers.

Production Honeypots:

- Deployed within an organization’s network to detect real-time threats.
- Provide early warning of potential attacks and help in fortifying defenses.

Functions of Honeypots

1. **Threat Detection**:
    - Identify unauthorized access attempts, malware infections, and insider threats.
2. **Attack Analysis**:
    - Collect detailed logs of attacker activities, tools, and methodologies.
3. **Decoy and Distraction**:
    - Divert attackers away from critical systems, buying time for defenders.
4. **Vulnerability Assessment**:
    - Understand which aspects of a system are most attractive to attackers.
5. **Training and Awareness**:
    - Serve as platforms for simulating attacks and training security teams.

Key Benefits

- **Minimal False Positives**: Honeypots only capture data when someone intentionally interacts with them, reducing noise.
- **Cost-Effectiveness**: Simpler honeypots can be inexpensive compared to complex intrusion detection systems (IDS).
- **Deep Insights**: Provide detailed logs and telemetry that might not be available through other security tools.
- **Support for Threat Intelligence**: Honeypots contribute data to threat intelligence databases.

Challenges and Risks

- **Risk of Exploitation**: If compromised, a honeypot could be used to launch attacks on other systems.
- **Detection by Attackers**: Skilled attackers may recognize honeypots and avoid or manipulate them.
- **Resource Intensive**: High-interaction honeypots require significant setup, monitoring, and management.
- **Legal and Ethical Issues**: Deploying honeypots might inadvertently expose sensitive data or raise privacy concerns.

Examples of Honeypot Tools

- **Kippo**: A low-interaction SSH honeypot.
- **Honeyd**: A tool for creating virtual honeypots.
- **Dionaea**: Focused on malware collection and analysis.
- **T-Pot**: A multi-honeypot platform offering integration with several honeypot technologies.
- **Canary Tokens**: Lightweight decoys used for detecting unauthorized access.

Best Practices for Deploying Honeypots

- **Segregation**: Isolate honeypots from production environments to prevent lateral movement if compromised.
- **Logging and Monitoring**: Implement robust logging to analyze activity and learn from attacks.
- **Realism**: Make honeypots appear authentic to entice attackers.
- **Regular Updates**: Maintain the honeypot with updated vulnerabilities to attract current threats.
- **Legal Compliance**: Ensure deployment adheres to legal and ethical guidelines in your jurisdiction.

Latest Trends in Honeynet Technology

Integration with Machine Learning:

- ML algorithms to analyze attack patterns and predict potential vulnerabilities.
- This helps in identifying unknown threats and zero-day attacks more effectively.

Specialized Honeynets for IoT and IIoT:

- Honeynets tailored to these environments are becoming popular.
- Mimic smart devices and industrial setups, attracting attackers

High-Interaction Honeynets:

- Allowing attackers to interact more deeply with decoy systems.
- Gathers detailed information on attack tools, tactics, and motivations, providing insights into both technical and behavioral aspects of attackers

Cloud-Based Honeynets:

- These setups can scale dynamically and are particularly useful for studying attacks on cloud services and infrastructure

Use in Active Defense Strategies:

- They not only gather information but also slow down and misdirect attackers, buying defenders time to respond

Honeynets for Critical Infrastructure:

- Cyber-physical systems, such as those used in utilities and transport, are adopting honeynets to detect and analyze threats targeting critical infrastructure.
- These setups are vital for preventing large-scale disruptions

Research on Honeynet Technology

https://www.honeynet.org/about/

https://www.honeynet.org/challenges/

https://www.codeproject.com/?cat=1

https://isc.sans.edu/

https://isc.sans.edu/tools/

https://www.backtrack-linux.org/

https://www.exploit-db.com/