Threat Intelligence Implementation Report

Introduction

This report details the practical implementation of threat intelligence principles, including the analysis of Indicators of Compromise (IoCs) and the setup and utilization of the OpenCTI Threat Intelligence Platform.

Part 1: Analysis of Indicators of Compromise (IoCs)

1.1 IoC #1: Malicious Domain

- **IoC:** A suspicious domain, "example.com," is identified in network traffic logs.
- **Detection Method:** Network intrusion detection systems (NIDS) and firewall logs are used to detect access to or communication with "example.com." DNS logs are analyzed for unusual domain resolution requests.
- **Indication of Threat:** The domain is linked to known phishing campaigns. Access to this domain may indicate a user falling victim to a phishing attack, potentially leading to credential theft or malware infection.

1.2 IoC #2: File Hash

- **IoC:** A specific file hash (e.g., SHA256: a1b2c3d4e5f6...) is observed on an endpoint.
- **Detection Method:** Endpoint detection and response (EDR) solutions are used to scan files and compare their hashes against a database of known malicious file hashes.
- **Indication of Threat:** The file hash matches that of a known ransomware executable. The presence of this file indicates a potential ransomware infection on the endpoint.

Part 2: OpenCTI Threat Intelligence Platform Implementation

2.1 OpenCTI Platform Setup

- **Implementation Method:** Docker Compose was used for the OpenCTI platform setup.
- **Documentation:**
    - A `docker-compose.yml` file was created to define the OpenCTI services (web, database, etc.).
    - The file includes configurations for services, ports, volumes, and dependencies.
    - Evidence: The provided document includes snippets of a `docker-compose.yml` file.

2.2 Connector Integration

- **Connector 1:** MISP Connector
    - Configuration details for the MISP connector were added to OpenCTI to ingest threat intelligence data from a MISP instance.
    - Documentation includes steps to configure the connector within the OpenCTI platform, providing the MISP instance URL and authentication key.
- **Connector 2:** VirusTotal Connector
    - The VirusTotal connector was configured to automatically enrich observables (file hashes, IPs, domains) within OpenCTI with data from VirusTotal.
    - Configuration steps involve providing the VirusTotal API key within the OpenCTI connector settings.
- **Documentation of Integration:** Documentation was created, detailing the installation process, including any necessary environment variables, network configurations, and authentication settings for each connector.

2.3 Basic Usage Demonstration

- **Data Ingestion:** Demonstrated the successful ingestion of threat intelligence data from the configured connectors (MISP and VirusTotal) into the OpenCTI platform.
- **Search and Analysis:** Showcased the ability to search for and analyze IoCs within OpenCTI, including viewing details about objects, relationships, and associated threat information.
- **Visualization:** Demonstrated the visualization of threat data within OpenCTI through graphs or charts, illustrating relationships between threat actors, IoCs, and campaigns.

Risk Management Strategies Report

This report outlines the application of risk management strategies, including the identification of risks from vulnerability scan results, treatment recommendations, mitigation steps, and a risk monitoring procedure.

1. Risk Assessment

Vulnerability Scan Results (Based on Provided Documents)

The vulnerability scan results, primarily from the Nmap scans, indicate the following:

- **Open Ports with Unknown Services:** The scan identified open TCP ports 51692 and 64660, running unknown services.

- **Potential Vulnerability: broadcast-avahi-dos:** The scan detected the presence of "After NULL UDP avahi packet DoS," indicating a potential denial-of-service vulnerability (CVE-2011-1002).
- **dnsmasq Vulnerability:** The target host is running dnsmasq 2.90, which has known vulnerabilities (e.g., CVE-2017-14493).

2. Identification of Critical Risks

Based on the vulnerability scan results, the following are identified as critical risks:

2.1 Critical Risk #1: Denial of Service (DoS) Attack via broadcast-avahi-dos

- **Explanation:** The "broadcast-avahi-dos" vulnerability allows an attacker to potentially crash systems on the local network by sending a crafted UDP packet. This can disrupt network services and cause significant downtime.
- **Likelihood:** Medium. An attacker on the local network could exploit this vulnerability.
- **Impact:** High. Successful exploitation can lead to service disruption, data loss, and reputational damage.

2.2 Critical Risk #2: Remote Code Execution via dnsmasq Vulnerability

- **Explanation:** dnsmasq is a widely used DNS and DHCP server. Vulnerabilities like CVE-2017-14493 can allow an attacker to execute arbitrary code on the affected system, potentially leading to full system compromise.
- **Likelihood:** Medium. Exploits for known dnsmasq vulnerabilities are publicly available.
- **Impact:** High. Successful exploitation can lead to complete loss of confidentiality, integrity, and availability of the affected system and potentially the entire network.

3. Treatment Recommendations

3.1 Treatment for DoS Attack via broadcast-avahi-dos

- **Recommendation:** Disable the Avahi service if it is not required. If required, apply patches to mitigate the vulnerability. Implement network segmentation to limit the impact of a successful attack.
- **Justification:** Disabling the service eliminates the vulnerability. Patching and network segmentation reduce the risk and impact of a potential attack.

3.2 Treatment for Remote Code Execution via dnsmasq Vulnerability

- **Recommendation:** Update dnsmasq to the latest version. Implement network access controls to limit access to the dnsmasq service.
- **Justification:** Updating to the latest version patches the vulnerability. Network access controls reduce the attack surface and limit the potential for exploitation.

4. Basic Mitigation Steps

4.1 Mitigation Steps for DoS Attack via broadcast-avahi-dos

1. Identify affected systems.
2. Disable the Avahi service on non-essential systems:
    - Linux: sudo systemctl stop avahi-daemon
    - Linux: sudo systemctl disable avahi-daemon
3. Apply patches for Avahi:
    - Use the system's package manager (e.g., sudo apt update && sudo apt upgrade).
4. Implement network segmentation using VLANs and firewalls.
5. Monitor network traffic for suspicious Avahi activity.

4.2 Mitigation Steps for Remote Code Execution via dnsmasq Vulnerability

1. Identify systems running dnsmasq.
2. Update dnsmasq to the latest version:
    - Debian/Ubuntu: sudo apt update && sudo apt upgrade dnsmasq
    - Red Hat/CentOS: sudo yum update dnsmasq
3. Configure firewall rules to restrict access to port 53 (DNS) and 67 (DHCP) to authorized systems.
4. Implement intrusion detection and prevention systems (IDS/IPS) to detect and block potential exploitation attempts.
5. Regularly audit and review dnsmasq configurations.

5. Risk Monitoring Procedure

Risk Monitoring Procedure: Critical System Vulnerability Monitoring

- **Objective:** To continuously monitor the identified critical risks (DoS via broadcast-avahi-dos and RCE via dnsmasq) and ensure that mitigation steps remain effective.
- **Procedure:**
    1. **Asset Inventory:** Maintain an up-to-date inventory of all systems, including their operating systems, services, and software versions.
        - **Tool:** Utilize a network scanning tool (e.g., Nmap) to periodically scan the network and update the asset inventory.
        - **Frequency:** Quarterly.
        - **Justification:** Ensures all assets are accounted for and their configurations are known.
    2. **Vulnerability Scanning:** Conduct regular vulnerability scans to identify new vulnerabilities and verify that existing vulnerabilities are patched.
        - **Tool:** Utilize a vulnerability scanner (e.g., Nessus, OpenVAS).
        - **Frequency:** Monthly.
        - **Justification:** Proactively identifies new vulnerabilities that may affect the systems.
    3. **Patch Management:** Track the availability of security patches for all systems and software.
        - **Tool:** Use a patch management system (e.g., WSUS, Ansible).
        - **Frequency:** Weekly.
        - **Justification:** Ensures that systems are patched in a timely manner.
    4. **Intrusion Detection:** Monitor network and system logs for signs of exploitation attempts.
        - **Tool:** Utilize an intrusion detection system (IDS) (e.g., Snort, Suricata) and a security information and event management (SIEM) system (e.g., Splunk, ELK stack).
        - **Frequency:** Continuously.
        - **Justification:** Provides real-time detection of potential attacks.

5. **Security Audits:** Conduct periodic security audits to review system configurations, access controls, and security policies.
    - **Tool:** Manual review and automated configuration assessment tools.
    - **Frequency:** Annually.
    - **Justification:** Verifies the effectiveness of security controls and identifies areas for improvement.
6. **Risk Review:** Review the identified risks and the effectiveness of mitigation steps.
    - **Participants:** IT security team, system administrators, and management.
    - **Frequency:** Quarterly.
    - **Justification:** Ensures that the risk management strategy remains appropriate and effective.

- **Documentation:** All monitoring activities, findings, and corrective actions will be documented in a risk management system.
- **Escalation:** Any identified critical risks will be escalated to the IT security manager and senior management within 24 hours.

Security Monitoring and Incident Response Report

This report details the setup of basic security monitoring, a use case demonstrating detection rules, alert prioritization, and response procedures, and an incident response scenario.

Part 1: Security Monitoring

1.1 Basic Security Monitoring Setup

Security monitoring involves the continuous collection and analysis of data to detect potential security threats. A basic setup includes the following components:

- **Log Management:** Centralized collection and storage of logs from various sources (servers, network devices, applications).
    - **Tool:** Syslog server (e.g., rsyslog, syslog-ng)
    - **Configuration:**

        - Install and configure a Syslog server on a dedicated system.
        - Configure devices (e.g., servers, firewalls, routers) to forward logs to the Syslog server using the appropriate protocol (e.g., UDP, TCP). For example, in Linux, you might edit /etc/rsyslog.conf or create a new configuration file in /etc/rsyslog.d/.
        - Example rsyslog.conf snippet:
          . @syslog_server_ip:514 # Send all logs to the Syslog server
        -
        -
        - Ensure the Syslog server has sufficient storage and log rotation policies to manage the volume of logs.
    - **Evidence of Functionality:**
        - Verify that logs are being received by the Syslog server. Check the Syslog server's log files (e.g., /var/log/syslog, /var/log/messages) for entries from the configured devices.
        - Use a command-line tool like tcpdump on the Syslog server to capture network traffic on the Syslog port (e.g., sudo tcpdump -i eth0 port 514) and confirm that log data is being transmitted.
- **Intrusion Detection System (IDS):** Monitors network traffic or system activity for malicious activity.
    - **Tool:** Snort (or Suricata)
    - **Configuration:**
        - Install Snort on a system connected to a network tap or span port to capture network traffic. This might involve installing Snort from a package manager (e.g., sudo apt install snort) or compiling from source.
        - Configure Snort to listen on the appropriate network interface and define the network traffic to be analyzed. This is done in the snort.conf file.
        - Obtain Snort rules (either from the Snort community or a commercial provider like Cisco Talos) and configure them in Snort. Rules define the attack patterns that Snort will detect.
        - Example snort.conf snippet (basic configuration):
          ipvar HOME_NET 192.168.1.0/24

        - ipvar EXTERNAL_NET any
        - portvar HTTP_PORTS 80
        -
        - ...
        -
        - include $RULE_PATH/snort.rules # Include the main Snort rules file
        -
        -
    - **Evidence of Functionality:**
        - Run Snort in test mode to verify that the configuration and rules are loaded correctly (e.g., sudo snort -T -c /etc/snort/snort.conf).
        - Generate test traffic that matches a Snort rule (e.g., by using a tool like curl or hping3 to send a specific attack pattern) and verify that Snort detects and logs the event. Check the Snort alert log file (e.g., /var/log/snort/alert).
        - Use tcpdump to verify that Snort is capturing network traffic.

- **Security Information and Event Management (SIEM):** Aggregates and analyzes security data from various sources to provide a centralized view of the security posture.
    - **Tool:** ELK Stack (Elasticsearch, Logstash, Kibana)
    - **Configuration:**
        - **Logstash:**
            - Install Logstash on a dedicated server.
            - Configure Logstash input plugins to receive logs from various sources (e.g., Syslog, file input for application logs).
            - Configure Logstash filter plugins to parse and structure the log data into a format that Elasticsearch can easily search (e.g., using grok filters to extract fields from unstructured log messages).
            - Configure Logstash output plugins to send the processed data to Elasticsearch.
        - Example Logstash configuration file (/etc/logstash/conf.d/syslog.conf):
          input {
        - syslog {

```
    port => 514
  }
 }
 filter {
  grok {
    match => { "message" => "%{SYSLOGTIMESTAMP:timestamp}
%{HOST:host} %{GREEDYDATA:message}" }
  }
 }
 output {
  elasticsearch {
    hosts => ["elasticsearch_server_ip:9200"]
    index => "syslog-%{+YYYY.MM.dd}"
  }
 }
```

- **Elasticsearch:**
  - Install Elasticsearch on one or more servers.
  - Configure Elasticsearch to store and index the data received from Logstash. This includes setting up cluster configuration, defining indices, and managing data storage.
- **Kibana:**
  - Install Kibana on a server.
  - Configure Kibana to connect to the Elasticsearch cluster.
  - Define index patterns in Kibana to access the data in Elasticsearch.
  - Create visualizations (e.g., charts, graphs, dashboards) to analyze and monitor the security data.

- **Evidence of Functionality:**
  - Verify that Logstash is receiving logs and sending them to Elasticsearch. Check the Logstash logs for any errors.
  - Verify that Elasticsearch is receiving and indexing the data. Use the Elasticsearch API (e.g., with curl) to query the indices and confirm that data is present. For example: curl -XGET "http://elasticsearch_server_ip:9200/_cat/indices?v"

- In Kibana, create an index pattern and verify that you can see the log data. Create a simple visualization (e.g., a histogram of log messages over time) to confirm that Kibana is working correctly.

1.2 Use Case: Detecting a Brute-Force Attack

1.2.1 Detection Rules

A brute-force attack involves an attacker attempting to gain unauthorized access to a system by repeatedly trying different username and password combinations. We'll focus on detecting this against an SSH server.

- **Detection Rule (Snort):**
  alert tcp any any -> any 22 (msg:"SSH Brute-Force Attack"; flow:established,to_server; content:"|50 41 53 53 57 4f 52 44|"; distance:0; within:8; threshold:type limit, track by_src, count 10, seconds 60; classtype:bad-unknown; sid:1000001; rev:1;)

-
-
    - **Explanation:**
        - alert tcp any any -> any 22: This specifies that Snort should inspect TCP traffic going to port 22 (SSH).
        - msg:"SSH Brute-Force Attack": This is the message that will be logged when the rule is triggered.
        - flow:established,to_server: This ensures that the rule only applies to established connections going to the server.
        - content:"|50 41 53 53 57 4f 52 44|": This looks for the ASCII representation of the string "PASSWORD" in the packet, which is often present in failed SSH login attempts. The | symbols indicate that the content is in hexadecimal format.
        - distance:0; within:8: These options specify where in the packet Snort should look for the content.
        - threshold:type limit, track by_src, count 10, seconds 60: This is the key part of the rule for detecting brute-force attacks. It tells Snort to track the number of times the specified content is seen from the same source IP address (track by_src). If the count reaches 10 within 60 seconds, the rule is triggered.
        - classtype:bad-unknown: This assigns a classification to the alert.

        - sid:1000001; rev:1;: These are the Snort ID and revision number for the rule.

- **Detection Rule (Elasticsearch/Kibana):**
    - **Index Pattern:** Define an index pattern in Kibana to match the SSH authentication logs (e.g., syslog-*). Ensure that your Logstash configuration is parsing the SSH logs and including relevant fields like timestamp, host, src_ip, user, and message.
    - **Query:** This query uses Kibana's query language (KQL) to find source IPs that have generated more than 10 failed SSH login attempts within a 60-second window.
      GET syslog-*/_search
    - {
    - "size": 0,
    - "query": {
    - "bool": {
    - "filter": [
    - {
    - "term": {
    - "program": "sshd" // Filter for SSH logs (program name might vary)
    - }
    - },
    - {
    - "match": {
    - "message": "Failed password" // Or a similar message indicating failed login
    - }
    - },
    - {
    - "range": {
    - "@timestamp": {
    - "gte": "now-60s",
    - "lte": "now"
    - }
    - }
    - }
    -]
    - }
    - },
    - "aggs": {
    - "attacking_ips": {
    - "terms": {
    - "field": "src_ip", // Or the field containing the source IP address

```
            "min_doc_count": 10
          }
        }
      }
    }
```

- **Explanation:**
  - The query filters the logs for SSH logs (program: "sshd") containing a message indicating a failed password attempt (message: "Failed password"). The exact message might vary depending on the SSH server configuration.
  - It also filters for events within the last 60 seconds (@timestamp).
  - The aggs section groups the results by the source IP address (src_ip) and counts the number of failed login attempts for each IP. The min_doc_count: 10 parameter ensures that only IPs with 10 or more failed attempts are returned.

- **Visualization:** In Kibana, create a visualization (e.g., a table) based on this query to display the list of attacking IPs. You can also set up an alert in Kibana to notify you when the number of attacking IPs exceeds a threshold.

1.2.2 Alert Prioritization Process

When a potential brute-force attack is detected, it's crucial to prioritize the alert to ensure a timely and effective response. Here's a sample prioritization process:

1. **Initial Detection:** The Snort rule or Kibana query triggers an alert.
2. **Automatic Enrichment:** The SIEM system (Kibana, in this case) automatically enriches the alert with additional information, such as:
   - Source IP address geolocation (to identify the origin of the attack).
   - The number of failed login attempts.
   - The target username (if available in the logs).
   - The affected system.

3. **Severity Assessment:** The SIEM system assigns a severity level to the alert based on predefined criteria:
    - **Critical:** More than 20 failed login attempts in 60 seconds, or if the target username is a privileged account (e.g., "root", "admin").
    - **High:** 10-20 failed login attempts in 60 seconds.
    - **Medium:** 5-9 failed login attempts in 60 seconds.
    - **Low:** Less than 5 failed login attempts in 60 seconds. This could be normal user error.
4. **Notification:** The SIEM system notifies the security team based on the severity level:
    - **Critical/High:** Immediate notification via multiple channels (e.g., phone call, SMS, email).
    - **Medium:** Email notification.
    - **Low:** Logged for future analysis.
5. **Escalation:** If the initial responder does not acknowledge the alert within a specified time (e.g., 5 minutes for critical alerts), the alert is escalated to a higher-level responder or on-call manager.

1.2.3 Response Procedures

The response to a detected brute-force attack depends on the severity of the alert. Here are the general procedures:

- **Low Severity:**
    - Monitor the activity for any increase in failed login attempts.
    - Review the logs to identify the user and system involved.
    - Consider implementing user training to prevent accidental lockouts.
- **Medium Severity:**
    - Investigate the source IP address using threat intelligence sources (e.g., VirusTotal, AbuseIPDB) to determine if it is associated with known malicious activity.
    - Temporarily block the source IP address at the firewall or IDS.

    - Notify the user whose account is being targeted.
    - Force a password reset for the affected user.
- **High Severity:**
    - Immediately block the source IP address at the firewall and IDS.
    - Isolate the affected system from the network to prevent further damage.
    - Initiate a more thorough investigation to determine the attacker's intent and any potential damage. This may involve examining system logs, file integrity, and running memory analysis.
    - Consider involving law enforcement if the attack is sophisticated or causes significant damage.
- **Critical Severity:**
    - Follow all the steps for High Severity.
    - Activate the incident response plan.
    - Gather a team of security experts to handle the incident.
    - Document all actions taken.
    - Prepare for potential data loss or system downtime.
    - Communicate with stakeholders as needed.

Part 2: Incident Response

2.1 Incident Response Scenario: Ransomware Attack

2.1.1 Classification of Incident

- **Incident Type:** Ransomware Attack
- **Description:** A workstation on the corporate network was infected with ransomware. The user reported that their files were encrypted and a ransom note appeared on their screen, demanding payment in cryptocurrency for decryption.
- **Impact:**
    - **Confidentiality:** High (files were encrypted).
    - **Integrity:** High (files were modified).

- **Availability:** High (files are inaccessible).

- **Severity:** Critical

2.1.2 Response Steps Taken

1. **Detection and Containment:**
    - The user reported the incident to the IT help desk, who escalated it to the security team.
    - The security team confirmed the ransomware infection by examining the user's workstation remotely.
    - The infected workstation was immediately isolated from the network to prevent the ransomware from spreading to other systems. This involved disabling the network interface on the workstation.
    - The security team checked network shares and other connected systems for any signs of lateral movement.
2. **Investigation:**
    - The security team obtained a copy of the ransomware sample from the infected workstation for analysis.
    - The ransomware was analyzed to identify the type of ransomware, the encryption algorithm used, and any potential weaknesses.
    - System logs on the infected workstation and network devices were examined to determine the initial infection vector (e.g., phishing email, malicious website).
    - The security team checked for any backups of the encrypted files.
3. **Eradication:**
    - The infected workstation was reimaged to ensure that the ransomware was completely removed.
    - Security patches were applied to all systems to address any vulnerabilities that may have been exploited.
    - Antivirus definitions were updated on all systems.
4. **Recovery:**
    - The encrypted files were restored from a recent backup.

- The restored files were verified to ensure their integrity.
- The user was provided with a new, secure workstation and instructed on how to avoid future ransomware infections.

5. **Post-Incident Activity:**
    - The incident was fully documented, including the timeline of events, the actions taken, and the results.

2.1.3 Lessons Learned

- **Importance of Backups:** The incident highlighted the critical importance of having regular, reliable backups of all critical data. The organization was able to recover from the attack without paying the ransom because they had recent backups.
- **Security Awareness Training:** The incident underscored the need for ongoing security awareness training for all employees. The training should cover topics such as how to identify phishing emails, how to avoid malicious websites, and the importance of not downloading attachments from untrusted sources.
- **Incident Response Plan:** The incident demonstrated the value of having a well-defined and tested incident response plan. The plan helped the security team to respond quickly and effectively to the attack, minimizing the damage.
- **Vulnerability Management:** The incident highlighted the importance of keeping all systems patched and up-to-date. Exploiting known vulnerabilities is a common way for ransomware to spread.
- **Network Segmentation:** The isolation of the infected workstation was effective in preventing the spread of the ransomware. The incident reinforced the importance of network segmentation to limit the impact of security incidents.
- **SIEM Effectiveness:** A properly configured SIEM can help in detecting and responding to ransomware attacks.

This report provides a detailed overview of security monitoring and incident response, including practical examples and recommendations.