Zeek

## **Zeek – TryHackMe Room Notes**

## 🔰 **Task 1: Introduction**

* **Zeek (formerly Bro)** is a powerful network analysis framework.
* Primarily used for **network security monitoring (NSM)**.
* It passively monitors traffic and generates logs, alerts, and insights.

## 🛡️ **Task 2: Network Security Monitoring and Zeek**

* **NSM** is the collection, analysis, and escalation of network traffic.
* Zeek differs from signature-based IDS (like Snort); it's more like a **network log generator & analyzer**.
* It inspects protocols and behaviors, not just known signatures.
* Zeek uses **event-driven scripting** for deeper inspection.

## 📁 **Task 3: Zeek Logs**

Zeek generates multiple logs depending on traffic type. Common logs:

* **conn.log**: General connection data (src/dst IPs, ports, protocols).
* **http.log**: HTTP requests/responses.
* **dns.log**: DNS queries.
* **files.log**: Extracted files and their metadata.
* **ssl.log**: TLS handshake metadata.

Each log has structured fields and timestamps.

## 🧰 **Task 4: CLI Kung-Fu Recall: Processing Zeek Logs**

* Zeek logs are in **TSV** format (tab-separated).
* Useful commands:
  + cat <logfile> | zeek-cut <field1> <field2> – extract fields.
  + sort, uniq, grep – process logs.
  + jq or awk – for scripting and automation.

Example:

bash

CopyEdit

cat conn.log | zeek-cut id.orig\_h id.resp\_h proto

## 🧪 **Task 5: Zeek Signatures**

* Zeek can **detect patterns using signatures**, similar to Snort.
* Signatures match on:
  + Strings in payloads.
  + Protocol behavior.
* Signatures are defined in .sig files and require Signature Framework.

Example:

zeek

CopyEdit

signature example {

ip-proto == tcp

payload /evilstring/

event "Malicious payload detected"

}

## 📜 **Task 6: Zeek Scripts | Fundamentals**

* Zeek uses **.zeek scripts** to define detection logic.
* Scripts use an **event-driven model** (triggered by traffic events).
* Basic script structure:

zeek

CopyEdit

event http\_request(c: connection, method: string, uri: string) {

print fmt("HTTP request: %s %s", method, uri);

}

* Scripts can **generate logs, raise notices, or trigger actions**.

## 🧩 **Task 7: Zeek Scripts | Scripts and Signatures**

* Combine **signatures + scripting** for powerful detection.
* Signature triggers can call **custom events** in scripts.
* Example: raise alert when a signature matches, then log context or take action.

## ⚙️ **Task 8: Zeek Scripts | Frameworks**

* Zeek has several built-in frameworks:
  + **Notice Framework**: for generating alerts.
  + **Intel Framework**: for matching IPs/domains against threat intel feeds.
  + **Input Framework**: for reading input data like CSV/intel files.
  + **Logging Framework**: custom log generation.

Example: Use the **Intel Framework** to detect known bad IPs/domains.

## 📦 **Task 9: Zeek Scripts | Packages**

* Zeek has a **package manager (zkg)** for managing community scripts.
* Use to install detection scripts, frameworks, analyzers.
* Common commands:
  + zkg install <package-name>
  + zkg list
  + zkg refresh

Examples of popular packages:

* zeek/community-id
* zeek/ja3
* zeek/mitrecor

## ✅ **Task 10: Conclusion**

* Zeek is **extensible, powerful**, and used widely in SOCs and threat hunting.
* Not a typical IDS – it's more of a **network behavioral monitoring** and analysis tool.
* Key strengths:
  + Deep protocol analysis.
  + Script-based detection.
  + Customizable logs and alerts.

