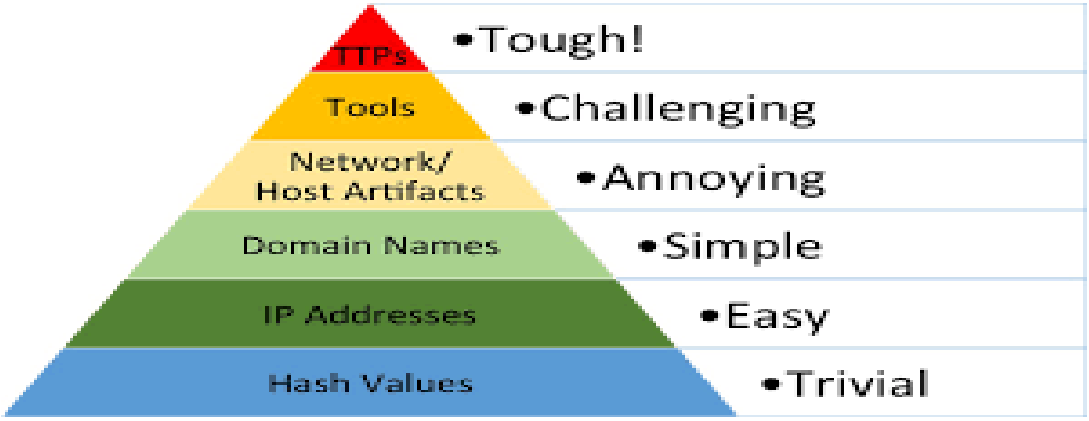


# Pyramid of Pain — Indicators Explanation

It shows how much **“pain” (difficulty)** you cause an **attacker when you detect or block their indicators.**

1. **HASH VALUES : (Trivial)**
2. **IP Address (Easy)**
3. **Domain Names (Simple)**
4. **Host Artifacts (Annoying)**
5. **Network Artifacts (Annoying)**
6. **Tools (Challenging)**
7. **TTP (Tough)**

---

# Hash Values

- Hash = **fixed-length fingerprint of data** produced by a **hashing algorithm (MD5, SHA-1, SHA-2, etc.).**
- Even **one bit** change → **completely different hash.**

- Reminder: appending even **small text or changing one bit** will produce a **different digest — hashes are content-sensitive.**

## Hash Lookups

- Using **hash lookups (like VirusTotal or MetaDefender)** is a quick way to identify **known malware** because you can check if a file's hash **matches a known malicious sample in threat intelligence databases.**

- Your **file's hash** matches a **virus hash in the database** only if the file's content is **identical** to a known **malware sample already stored there.**

- Even a **1-byte change alters the hash completely** — which is why this method works only for **exact matches.**

- However, it's not sufficient alone because:

    - Attackers can easily modify files **(even a single byte), changing the hash and evading detection.**

    - Hash-based detection only works for known threats, not new or polymorphic malware.

- Use multi-source threat intel (VirusTotal, MetaDefender, vendor EDR telemetry) — correlate hashes with behavior and metadata.

## Fuzzy Hashing

- Fuzzy hashing means it checks **how similar two files' contents are (like a percentage match)**, instead of **requiring an exact byte-for-byte match.**

- So even if an **attacker slightly changes the file**, **ssdeep can still recognize it as a similar variant.**

# IP Addresses as Indicators

- **IPs identify** devices on a network and are **quick to block or monitor** (useful short-term detection).
- Limitation: **attackers can easily change IPs** (new hosts, proxies, cloud services), so blocking **IPs is fragile and won't stop determined actors.**

---

# Fast Flux

- **Fast Flux** = a **DNS trick** where a **domain rapidly rotates many compromised** hosts' IPs (short DNS TTLs) **so the same domain resolves to different IPs constantly.**
- Fast Flux is a DNS technique used by **botnets to hide phishing**, web proxying, malware delivery, and malware communication a**ctivities behind compromised hosts acting as proxies.**

### TTL

- TTL (Time To Live) is a value (in seconds) set on a DNS record that tells resolvers and **caches how long they should keep that DNS answer before asking the authoritative server again.**
- **Example: TTL = 86400 → cache for 24 hours.**
- **Short TTLs (like 30, 60, or 300 seconds) force resolvers to re-query frequently.**

### How Fast Flux Works (Step-by-Step)

1. Attacker controls a domain and sets very short DNS TTLs.

2. The domain's DNS returns a pool of compromised machines (many different IPs).
3. Those compromised machines act as proxies or hosts for malicious content; the set of IPs keeps changing.
4. Observer blocks an IP → domain still works because other IPs respond.

### Detection Hints

- Domain with many **A records that change frequently**.
- Very **low DNS TTLs (seconds/minutes).**
- IPs that are end-user/residential or part of unexpected ASNs.
- Multiple **unrelated domains resolving to the same rotating IP set.**
- Unusual HTTP headers, repeated proxy patterns, or abnormal traffic timing.

# Domains

### Why domains hurt attackers more than IPs

- **Purchasing/registering and managing** domains **takes time, cost, and leaves an audit trail** (r**egistrar, WHOIS, payment, DNS records, APIs)**.
- Changing domains **requires new registrations or abuse of providers/APIs,** which raises **attacker overhead and exposure.**
- Effect on attacker ops: **they must obtain new domains, update DNS, propagate records, and update tooling/links —** all of which increase cost, time, and chance of mistakes that defenders can exploit.

---

# Punycode Attack

- Punycode is a system that converts **Unicode (non-English)** domain names into **ASCII (English letters and numbers)** — because **DNS only understands ASCII.**
- Unicode includes all **global characters (like ü, ñ, Cyrillic)**, but **DNS only understands ASCII (English letters, numbers, symbols).** Punycode converts Unicode domains into ASCII so DNS can process them, e.g., **münich.com** → **xn--mnich-kva.com**.
- **apple.com** (Cyrillic “a”) looks like **apple.com**, but actually becomes **xn--pple-43d.com** — a malicious site.

### Example

Unicode domain → **münich.com**
Punycode version → **xn--mnich-kva.com**

### Security Impact

- Attackers exploit this to visually mimic real domains using look-alike Unicode characters.
- Example:
    - Fake domain → **apple.com** (Cyrillic “a”, not English “a”)
    - Punycode → **xn--pple-43d.com**
- It looks like apple.com but is malicious.

Browsers like IE, Chrome, Edge, and Safari now translate obfuscated characters into the full Punycode name.

---

# URL Shorteners

- Attackers usually hide **malicious domains under URL shorteners.**
- A URL Shortener creates a s**hort and unique URL that redirects to the real malicious site.**

---

## ANY.RUN Sandbox Terminology

- Any.run is an **interactive cloud sandbox where you can upload a suspicious/malicious file and execute it** in an instrumented virtual machine.

### Sample

Suspicious/malicious file you want to analyze.

### Detonation

Executing the **suspicious file safely inside the sandbox** to **observe its behavior.**
Meaning: run the sample in an isolated environment to see what it does.

### Sandbox Evasion

If malware **can't access the internet or C2 server,** it may:

- stop execution
- hide behavior
- avoid revealing malicious actions

---

# Host Artifacts (Annoying)

- The attacker will feel **more annoyed and frustrated if you can detect the attack.**
- This is **very time-consuming** for the attacker, **requiring more resources and tool modification.**
- Host artifacts are traces/observables attackers leave on the system:
    - **registry values**
    - **suspicious process execution**
    - **attack patterns or IOCs**
    - **dropped files**
    - **anything exclusive to the current threat**

### Example

- Suspicious processes (e.g., **winword.exe starting powershell.exe**)

---

# Network Artifacts (Annoying)

- Network artifacts also belong to the yellow zone in the Pyramid of Pain.
- Detecting them **forces attackers to change tactics or modify tools.**

### Network Artifacts Are:

- Unusual **User-Agent strings**
- **C2 server connections**

- Suspicious **URI patterns**
- Repeated **HTTP POST requests**

### Detection Sources

- PCAP files (**Wireshark**, TShark)
- IDS logs (**Snort**, Suricata)

### Terms

- **User-Agent string** = browser/app identifier
- **URI** = part after domain in URL
    - Example:
        - **URL:** `https://example.com/login.php?id=123`
        - **URI:** `/login.php?id=123`

---

# Tools (Challenging)

- If you **detect and stop the tools**, attackers may **give up or must build new tools.**
- They need t**o invest money, find replacements, or get training to use new tools.**

### Defensive Tools

1. Antivirus signatures
2. Detection rules
3. YARA rules
    - 

**YARA Rules**

- Custom **rules to detect malware based on:**
    - **strings**
    - **file structure**
    - **behavior**
- Used to detect m**alware families or variants.**

---

# TTPs (Tactics, Techniques & Procedures)

- **Tactics = high-level goals** (Initial Access, Persistence, Exfiltration).
- **Techniques = how the goals are achieved** (Phishing, PowerShell, Scheduled Task).
- **Procedures = exact steps/commands/tool**s.