Pyramid of Pain – Notes for SOC Analysts (Including Task 2–8 Indicators)

Introduction to the Pyramid of Pain

The Pyramid of Pain is a cybersecurity model created by David J. Bianco that illustrates how different types of threat indicators vary in the level of difficulty they impose on an adversary when defenders detect and block them. The higher you go in the pyramid, the more “pain” you cause to attackers by forcing them to modify significant parts of their infrastructure, tools, or tactics.

The Pyramid Levels (bottom to top):

1. **Hash Values (Trivial)**

2. **IP Addresses (Easy)**

3. **Domain Names (Simple)**

4. **Host Artifacts (Annoying)**

5. **Network Artifacts (Annoying)**

6. **Tools (Challenging)**

7. **TTPs – Tactics, Techniques & Procedures (Tough)**

Blocking indicators at the top of the pyramid forces attackers to redesign their operations rather than just swap small pieces of infrastructure.

Task 2: Hash Values (Trivial)

Description:

Hash values (MD5, SHA1, SHA256) uniquely identify specific malicious files.

Blocking a hash is trivial for the attacker because they can easily recompile or modify a file to generate a new hash.

SOC Example:

- **SHA256:** `d2c7a8e9f...9af`

- Defender blocks hash in EDR → attacker rebuilds payload → new hash appears.

Real Incident Example:

A malicious DLL is detected and blocked. After 5 minutes, the attacker uploads a slightly modified DLL with an entirely new hash, bypassing the block.

Task 3: IP Addresses (Easy)

Description:

Attackers frequently rotate IP infrastructure. Changing IPs costs them little effort, especially if using VPS, proxies, or botnets.

SOC Example:

Block outbound connection to:

- `45.148.10.131` (from your earlier phishing incident)

Real Incident Example:

C2 server is blocked by firewall → attacker updates implant to contact a different IP → communication resumes.

Task 4: Domain Names (Simple)

Description:

Domains are slightly harder to replace than IPs because attackers must register new ones or use DGAs (Domain Generation Algorithms).

SOC Example:

Malicious domain:

- `m1crosoftsupport.co`

Real Incident Example:

Attacker uses brand impersonation domain. Defender blocks it. Attacker registers a new domain like:

- `microsoft-support-helpdesk.net`

Task 5: Host Artifacts (Annoying)

Description:

These are traces left on the endpoint—file paths, registry keys, scheduled tasks, services, mutexes.

These require attackers to rebuild or significantly modify malware.

Examples:

- File dropped at: `C:\Users\Public\svchost.exe`

- Registry persistence key:

`HKCU\Software\Microsoft\Windows\Run\Updater`

- Suspicious PowerShell command stores payload in `%TEMP%`.

Why “Annoying”?

Malware developers must rewrite parts of their loader or change persistence mechanisms.

Task 6: Network Artifacts (Annoying)

Description:

These include URI paths, user-agent strings, beacon intervals, SSL certificate anomalies, packet structures.

Examples:

- C2 beacon URL: `/api/v1/getCommands`

- Custom user-agent: `Mozilla/5.0 (BackdoorClient)`

- Unique JA3 fingerprint for malware HTTPS traffic.

Why “Annoying”?

Changing network behavior often requires rewriting the C2 framework.

Task 7: Tools (Challenging)

Description:

If defenders detect and block specific attacker tools, attackers must switch to new tools or modify existing ones.

Examples:

- Blocking **Cobalt Strike** beacons using SIGMA rules.

- Detecting **Mimikatz** execution via memory inspection.

- Disrupting **Metasploit** modules by signatures.

Why “Challenging”?

- Attackers must change framework configurations.

- They may need new licenses, cracked versions, or custom tooling.

Task 8: TTPs (Tough)

Description:

Tactics, Techniques, and Procedures describe *how* attackers operate.

Changing TTPs requires redesigning the entire kill chain.

Examples (mapped to MITRE ATT&CK;):

- **Initial Access:** Phishing with HTML smuggling
- **Execution:** PowerShell in-memory payload loading
- **Persistence:** Scheduled tasks or registry autorun
- **Lateral Movement:** SMB + stolen credentials
- **Exfiltration:** Using DNS tunneling

Why “Tough”?

You force attackers to abandon habitual methods and rethink their workflow.

Summary Table

Indicator Type	Difficulty	Impact on Attacker
Hashes	Trivial	Recompile file
IPs	Easy	Change server address
Domains	Simple	Register new domain
Host Artifacts	Annoying	Modify malware design
Network Artifacts	Annoying	Rewrite C2 behavior
Tools	Challenging	Replace core tooling
TTPs	Tough	Redesign attack operations

Why SOC Analysts Must Understand the Pyramid

- Improves incident response decisions
- Helps prioritize which indicators matter
- Supports building long-term detection strategy
- Encourages focusing on attacker behaviour, not just signatures

---

End of Notes.