Tracking Pyramid C2: Identifying Post-Exploitation Servers in Hunt

Published: 2025-02-12 · Archived: 2026-04-05 12:58:14 UTC

TABLE OF CONTENTS

Public code repositories like GitHub have made offensive security tools widely accessible, benefiting defenders and adversaries. Red team frameworks designed for **post-exploitation and stealthy execution** are frequently shared, making it easier for threat actors to repurpose them for malicious use. **Pyramid**, an open-source post-exploitation framework written in Python, is one such tool. Among its features is a **lightweight HTTP/S server** that delivers encrypted payloads, allowing attackers to execute tasks while blending into legitimate Python activity.

Identifying detection indicators within publicly available offensive security tools before they appear in real-world attacks is critical for defenders. While Pyramid is valuable for penetration testers, its features can also enable adversaries to establish **command-and-control (C2) infrastructure** with minimal detection. By analyzing its open-source code, defenders can create **robust network detections** to prevent its use in malicious campaigns.

This post examines **Pyramid's HTTP/S server**, outlines **network signatures for detection**, and highlights **recently identified servers using the tool**. As adversaries increasingly rely on open-source tooling, defenders must stay ahead by proactively developing detection strategies that adapt to evolving TTPs.

What is Pyramid?

First released on GitHub in 2023, Pyramid is an open-source post-exploitation server designed to evade endpoint detection and response (EDR) tools using Python's legitimate presence in many environments. The tool uses a Python-based HTTP/S server capable of delivering files, which acts as a command-and-control server for offensive/malicious operations.

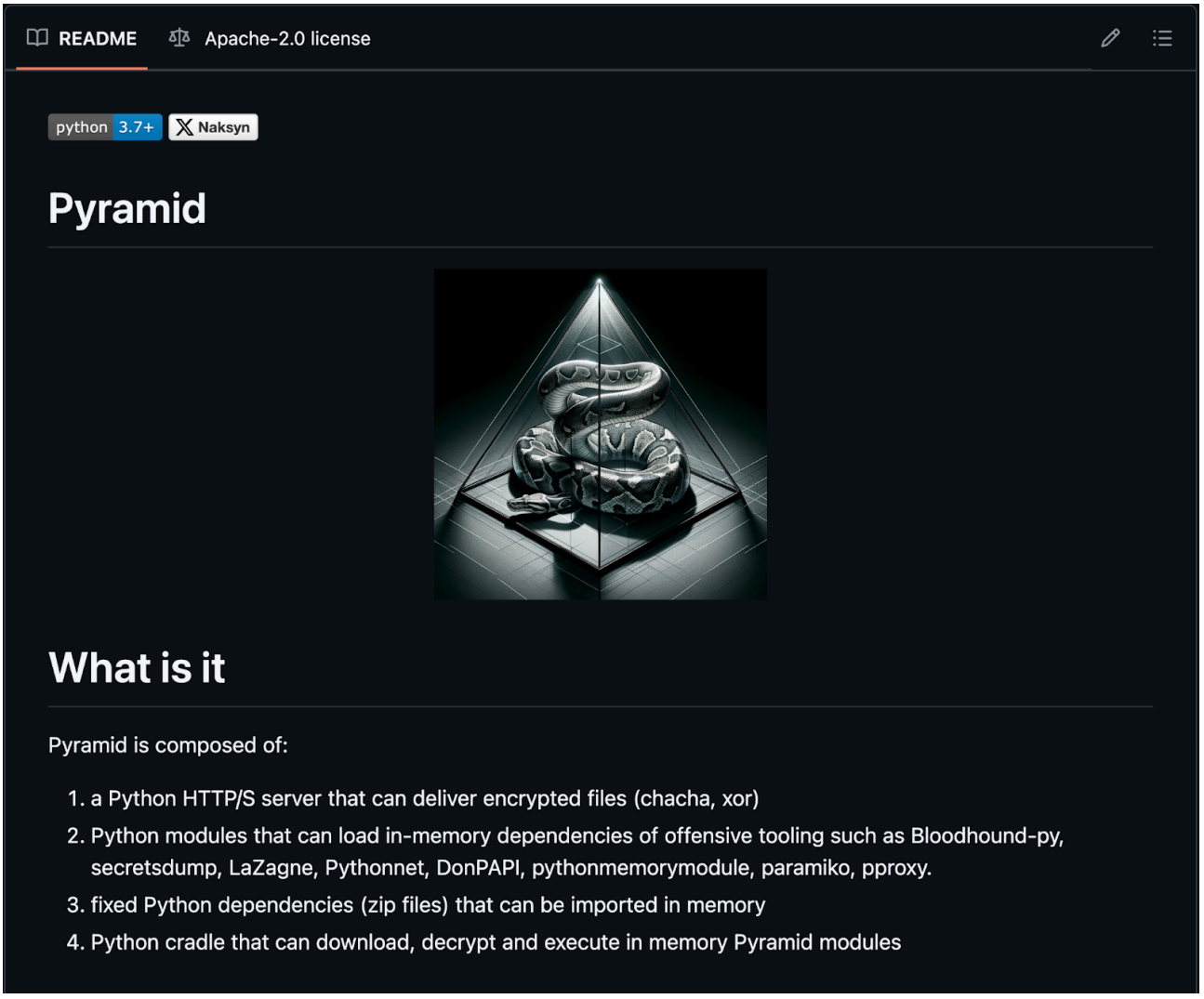

Figure 1: Screenshot of Pyramid README.

The project ships with several modules that load well-known programs like BloodHound, secretsdump, and LaZagne directly into memory. This in-memory execution allows operators to act within the context of a signed Python interpreter, a legitimate process that may slip past traditional endpoint security measures.

In December 2024, The DFIR Report published a blog post on TA4557 (also referred to as FIN6) activity leveraging Cobalt Strike and Pyramid to target job seekers with the more_eggs malware.

This past January, GuidePoint Security identified a campaign involving a RansomHub affiliate employing a Python-based backdoor to compromise endpoints. As will be discussed later, a number of the IP addresses in their IoC section overlapped with the infrastructure we detect as Pyramid.

Detection Opportunities

Analyzing the project's README reveals that the Pyramid agent communicates with its server over HTTP or HTTPS, using Basic HTTP authentication for access control. Given this, we can examine unique request-response patterns to aid in developing detection signatures.

Without valid credentials, we anticipate that, in addition to 200 OK responses, 401 Unauthorized responses will also appear in the relevant code. A simple search on GitHub confirms this expectation, revealing full HTTP headers that can help identify additional infrastructure.

```
123
124         def do_HEAD(self):
125                 self.send_response(200)
126                 self.send_header('Content-type', 'application/json')
127                 self.end_headers()
128
129         def do_AUTHHEAD(self):
130                 self.send_response(401)
131                 self.send_header(
132                         'WWW-Authenticate', 'Basic realm="Demo Realm"')
133                 self.send_header('Content-type', 'application/json')
134                 self.end_headers()
135
136         def do_GET(self):
137                 self.parsed_options=options
138                 key = self.server.get_auth_key()
139
140                 ''' Present frontpage with user authentication. '''
141                 if self.headers.get('Authorization') == None:
142                         self.do_AUTHHEAD()
143
144                         response = {
145                                 'success': False,
146                                 'error': 'No auth header received'
147                         }
148
149                         self.wfile.write(bytes(json.dumps(response), 'utf-8'))
150
```

Figure 2: Pyramid C2 HTTP 401 response.

When interacting with a suspected Pyramid server, the response headers exhibit the following characteristics:

```
  Server: BaseHTTP/0.6 Python/3.10.4\
Date:\
WWW-Authenticate: Basic realm="Demo Realm"\
Content-Type: application/json
```

Copy

Additionally, the server also returns a JSON response body:

```
{ "success": false, "error": "No auth header received" }
```

Copy

Figure 3 displays what the operator sees when navigating to the login page of a Pyramid server.

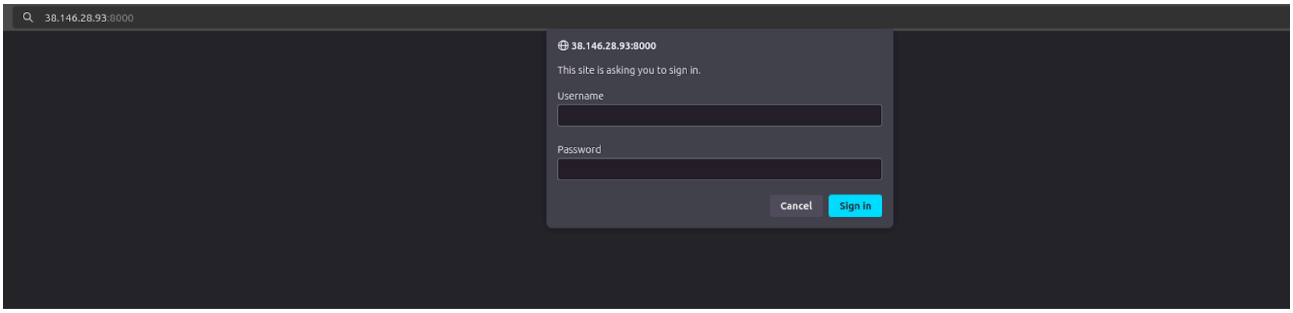

Figure 3: Screenshot of Pyramid authentication prompt.

With this information, we can immediately begin constructing a query to identify similar servers in the wild. To do this, we will leverage the following characteristics:

- **HTTP Status Code**: 401 Unauthorized
- **Response Body Hash**: SHA-256: 54477efe7ddfa471efdcc83f2e1ffb5687ac9dca2bc8a2b86b253cdbb5cb9c84
- **Server Header**: The project is based on Python 3.10 and uses BaseHTTP version 0.6. Since these values may vary slightly across deployments, we can generalize them as wildcards (BaseHTTP/0.* Python/3.*).
- **Authentication and Content Headers**: The presence of WWW-Authenticate: Basic realm="Demo Realm" and Content-Type: application/json.

By combining these elements, we can craft a structured query to identify Pyramid-related infrastructure:

```
http.status_code = "401" AND http.body.hash = "54477efe7ddfa471efdcc83f2e1ffb5687ac9dca2bc8a2b86b253cdbb5cb9c84" AND http.
```

Copy

This approach allows us to filter out unrelated infrastructure while maintaining flexibility for potential variations in server configurations. Refining these parameters further can improve detection fidelity and uncover additional instances of Pyramid servers deployed in the wild.

Using the parameters outlined above, we ran a search in Hunt, and our scans identified just 9 IP addresses across several different ports matching our criteria. The limited number of results further reinforces the specificity of our approach, suggesting that we are likely on the right track in identifying Pyramid-related infrastructure.

Beyond serving as an indicator of potential Pyramid servers, combining the JSON response and specific HTTP headers also serves as a validation mechanism. This is a critical step in crafting high-fidelity detection queries, as it helps reduce false positives. By ensuring that WWW-Authenticate: Basic realm="Demo Realm" and Content-

Type: application/json align with the expected JSON response body, we can more confidently filter out unrelated infrastructure while maintaining a precise detection signature.

Below is a snippet of the servers we track in Hunt under the Active C2s feature.

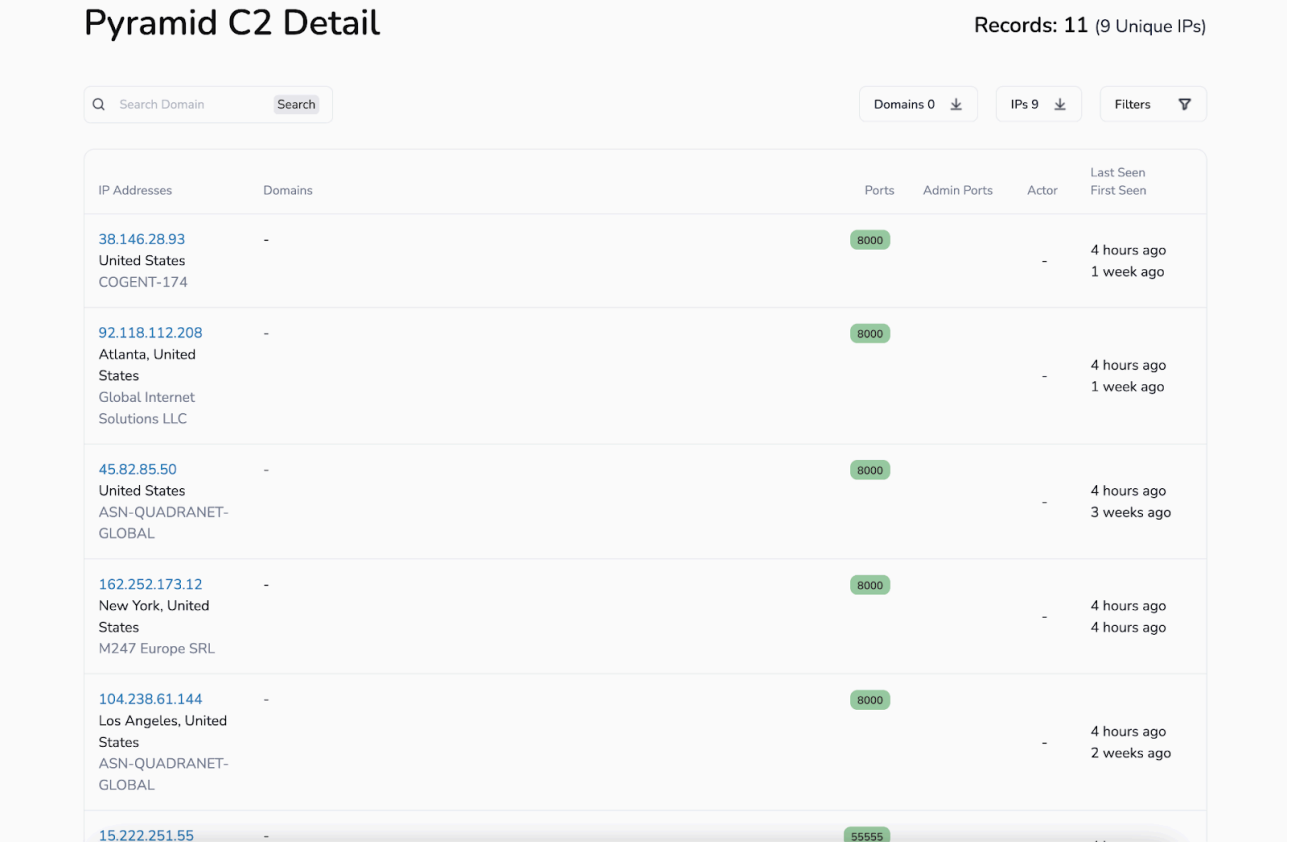

Figure 4: Snippet of the Pyramid C2 servers tracked in Hunt.

Recent Findings

Three of the IP addresses identified in our search---**104.238.61[.]144, 92.118.112[.]208, and 45.82.85[.]50**---were previously listed as indicators of compromise (IoCs) in **GuidePoint Security's** analysis of **RansomHub**. According to their report, these IPs were associated with a Python-based backdoor, suggesting a possible connection between this infrastructure and previously observed malicious activity.

One of the servers, **54.38.94[.]225**, resolves to multiple domains that bear a strong resemblance to DevaGroup, an internet marketing service based in Poland. While we have not yet identified any malicious samples associated with this server, the domains in question were recently registered in December 2024, indicating a possible attempt at phishing or drive-by downloads.

Conclusion

The infrastructure identified in this post exhibits distinctive HTTP response patterns, allowing for structured detection queries to surface additional related servers. This method will enable defenders to monitor or preemptively block suspect IPs before they are operationalized in attacks.

While open-source offensive security tools remain a common choice for adversaries, their deployment often leaves behind detectable network artifacts. Defenders can improve detection fidelity while minimizing false positives by focusing on authentication challenges, response headers, and specific error messages. Tracking similar implementations of these tools over time can provide early warning of new infrastructure and help refine detection methodologies as adversaries adapt.

Network Observables and Indicators of Compromise (IOCs)

IP Address	ASN	Domains
38.146.28[.]93	Cogent Communications	N/A
92.118.112[.]208	GLOBAL CONNECTIVITY SOLUTIONS LLP	N/A
162.252.172[.]12	M247 Europe SRL	N/A
104.238.61[.]144	GWY IT PTY LTD	N/A
45.82.85[.]50	QuadraNet Enterprises LLC	N/A
15.222.251[.]55	Amazon.com, Inc.	N/A
85.208.139[.]131	GLOBAL CONNECTIVITY SOLUTIONS LLP	N/A
54.38.94[.]225	OVH SAS	devagroup[.]com[.]pl **Subdomains:** www.aw. *pop.vps.* test.* smtp.panel.* smtp.backup.* serp.* mail.crm.* mail.git.* awscraper.* pop.ncrm.* pop.git.* ftp.* www.panel.* mail.aw.* www.parkwodny.* linki.* rentplanet.*
38.180.195[.]187	M247 Europe SRL	thiscode[.]info emdr-traumatherapie[.]info

---

Source: https://hunt.io/blog/tracking-pyramid-c2-identifying-post-exploitation-servers