GCP Threat Hunting Test Environment

Deployment Documentation

Version: 1.0 | Date: January 05, 2026 | Environment: GCP Instance (xdgaisocapp01)

GCP Threat Hunting Test Environment - Deployment Documentation

Last Updated: January 5, 2026 (Added testing results)

Python Version: 3.6.8

Table of Contents

Overview

This document describes the complete deployment of the Nextron-style Threat Hunting platform (THOR, ASGARD, VALHALLA) on a GCP instance. The deployment includes all three agents with Python 3.6 compatibility fixes.

Components Deployed

- **THOR Endpoint Agent** - Endpoint scanning and threat detection
- **ASGARD Orchestration Agent** - Fleet-wide campaign management
- **VALHALLA Feed Manager** - Threat intelligence feed aggregation

Deployment Summary

- **Instance:** xdgaisocapp01 (asia-southeast2-a)
- **Project:** chronicle-dev-2be9
- **Directory:** `~/threat-hunting-test/`
- **Python Environment:** Virtual environment (venv)
- **Status:** ✅ Fully Operational

System Requirements

GCP Instance Specifications

- **OS:** RHEL 8 (or compatible Linux)
- **Python:** 3.6.8
- **User:** app
- **Permissions:** sudo access for system package installation

System Packages Installed

```
python36-devel
yara-devel
yara
gcc
jansson-devel
protobuf
protobuf-compiler
```

Python Packages Installed

```
google-cloud-pubsub==2.13.0
google-cloud-firestore==2.5.3
google-cloud-bigquery==2.34.4
google-cloud-storage>=2.0.0,<2.11.0
google-cloud-compute>=1.0.0,<1.15.0
yara-python>=4.3.0,<4.6.0
psutil>=5.9.0
requests>=2.20.0,<2.31.0
dataclasses
typing_extensions>=3.7.4,<4.0.0
```

Deployment Steps

Step 1: Create Directory Structure

```
mkdir -p ~/threat-hunting-test
cd ~/threat-hunting-test
mkdir -p config logs data/yara_rules data/iocs data/sigma_rules scripts tests
docs
```

Step 2: Download Threat Hunting Files

Files were downloaded from GitHub repository:

```
cd ~/threat-hunting-test

# Main Python files
curl -f -s -o thor_endpoint_agent.py
https://raw.githubusercontent.com/ghifiardi/ai-driven-
soc/main/thor_endpoint_agent.py
curl -f -s -o asgard_orchestration_agent.py
https://raw.githubusercontent.com/ghifiardi/ai-driven-
soc/main/asgard_orchestration_agent.py
curl -f -s -o valhalla_feed_manager.py
```

```
https://raw.githubusercontent.com/ghifiardi/ai-driven-
soc/main/valhalla_feed_manager.py
curl -f -s -o threat_hunting_quickstart.py
https://raw.githubusercontent.com/ghifiardi/ai-driven-
soc/main/threat_hunting_quickstart.py
curl -f -s -o requirements_threat_hunting.txt
https://raw.githubusercontent.com/ghifiardi/ai-driven-
soc/main/requirements_threat_hunting.txt
curl -f -s -o THREAT_HUNTING_README.md
https://raw.githubusercontent.com/ghifiardi/ai-driven-
soc/main/THREAT_HUNTING_README.md

# Config files
curl -f -s -o config/thor_config.json
https://raw.githubusercontent.com/ghifiardi/ai-driven-
soc/main/config/thor_config.json
curl -f -s -o config/asgard_config.json
https://raw.githubusercontent.com/ghifiardi/ai-driven-
soc/main/config/asgard_config.json
curl -f -s -o config/valhalla_config.json
https://raw.githubusercontent.com/ghifiardi/ai-driven-
soc/main/config/valhalla_config.json
```

Step 3: Install System Packages

```
sudo yum install -y python3-devel yara-devel gcc
```

Step 4: Set Up Python Virtual Environment

```
cd ~/threat-hunting-test
python3 -m venv venv
source venv/bin/activate
pip install --upgrade pip
```

Step 5: Install Python Dependencies

```
# Install compatible versions for Python 3.6
pip install "requests>=2.20.0,<2.31.0"
pip install "yara-python>=4.3.0,<4.6.0"
pip install "psutil>=5.9.0"
pip install "google-cloud-pubsub==2.13.0"
pip install "google-cloud-firestore==2.5.3"
pip install "google-cloud-bigquery==2.34.4"
pip install "google-cloud-storage>=2.0.0,<2.11.0"
pip install "google-cloud-compute>=1.0.0,<1.15.0"
pip install "dataclasses"
pip install "typing_extensions>=3.7.4,<4.0.0"
```

Step 6: Apply Python 3.6 Compatibility Fixes

See [Python 3.6 Compatibility Fixes](#python-36-compatibility-fixes) section below.

Step 7: Update Configuration Files

```
cd ~/threat-hunting-test
bash scripts/update_config.sh
```

This updates all config files with GCP project ID: `chronicle-dev-2be9`

Python 3.6 Compatibility Fixes

Since the GCP instance runs Python 3.6.8, several compatibility fixes were required:

1. dataclasses Module

Issue: `dataclasses` was introduced in Python 3.7

Fix:

```
pip install dataclasses
```

2. asyncio.run() Method

Issue: `asyncio.run()` was introduced in Python 3.7

Fix: Replace in `valhalla_feed_manager.py` and `asgard_orchestration_agent.py`:

Before:

```
asyncio.run(main())
```

After:

```
loop = asyncio.get_event_loop()
loop.run_until_complete(main())
```

3. asyncio.create_task() Method

Issue: `asyncio.create_task()` was introduced in Python 3.7

Fix: Replace in `asgard_orchestration_agent.py`:

Before:

```
asyncio.create_task(self.execute_campaign(campaign_id))
```

After:

```
asyncio.ensure_future(self.execute_campaign(campaign_id))
```

4. langgraph Library

Issue: `langgraph` doesn't support Python 3.6

Fix: Make import optional in `thor_endpoint_agent.py` and `asgard_orchestration_agent.py`:

Before:

```
from langgraph.graph import StateGraph, END
```

After:

```
try:
    from langgraph.graph import StateGraph, END
except ImportError:
    StateGraph = None
    END = None
```

5. Missing Imports

Issue: Missing `os` module import

Fix: Add to `valhalla_feed_manager.py` and `asgard_orchestration_agent.py`:

```
import os
```

6. Package Version Compatibility

Issue: Some packages require newer Python versions

Fix: Install compatible versions:

- `requests>=2.20.0,<2.31.0` (instead of >=2.31.0)
- `google-cloud-pubsub==2.13.0` (instead of >=2.18.0)
- `google-cloud-firestore==2.5.3` (instead of >=2.11.0)
- `google-cloud-bigquery==2.34.4` (instead of >=3.11.0)

Component Status

✅ VALHALLA Feed Manager

Status: Operational

Initialization: Successful

Features Working:

- Feed manager initialization
- Configuration loading
- IOC feed structure (requires API keys for external feeds)

Known Limitations:

- External threat feeds require API keys (ThreatFox, MalwareBazaar)
- Some YARA rule URLs may need updating

Test Command:

```
cd ~/threat-hunting-test
source venv/bin/activate
python valhalla_feed_manager.py
```

✅ THOR Endpoint Agent

Status: Fully Operational

Features Working:

- Filesystem scanning

- Process scanning
- Network scanning
- IOC matching
- YARA rule scanning (when rules are available)

Test Results:

- Successfully scanned 3,061 files
- Scan duration: ~1 second
- No threats detected (expected in clean environment)

Test Commands:

```
# Quick filesystem scan
python thor_endpoint_agent.py --config config/thor_config.json --scan-type
filesystem --target /tmp

# Full scan with YARA and IOCs
python thor_endpoint_agent.py --config config/thor_config.json --scan-type full
--load-yara --load-iocs

# Process scan
python thor_endpoint_agent.py --config config/thor_config.json --scan-type
process
```

Expected Warnings (Non-Critical):

- Pub/Sub topic `thor-findings` doesn't exist (optional)
- BigQuery table doesn't exist (optional)
- YARA rules path doesn't exist (can be configured)

✅ ASGARD Orchestration Agent

Status: Operational

Features Working:

- Campaign creation
- Campaign execution
- Endpoint discovery (requires endpoint registration)
- Configuration management

Test Results:

- Successfully created campaign: "Ransomware Hunt Q4 2025"
- Campaign execution completed
- No targets selected (expected - no endpoints registered)

Test Command:

```
python asgard_orchestration_agent.py --help
```

Usage Guide

Activating the Environment

Always activate the virtual environment before use:

```
cd ~/threat-hunting-test
source venv/bin/activate
```

Running THOR Scans

Basic filesystem scan:

```
python thor_endpoint_agent.py --config config/thor_config.json --scan-type
filesystem --target /path/to/scan
```

Full scan with all features:

```
python thor_endpoint_agent.py --config config/thor_config.json --scan-type full
--load-yara --load-iocs
```

Process scan:

```
python thor_endpoint_agent.py --config config/thor_config.json --scan-type
process
```

Using VALHALLA

Initialize and update feeds:

```
python valhalla_feed_manager.py
```

Add custom YARA rules:

9. Place `.yar` files in `data/yara_rules/`
10. VALHALLA will load them automatically

Using ASGARD

View help:

```
python asgard_orchestration_agent.py --help
```

Create a campaign:

The agent includes example campaign creation in its main function.

Troubleshooting

Issue: ModuleNotFoundError

Solution: Install missing package:

```
source venv/bin/activate
pip install <package_name>
```

Issue: YARA compilation errors

Solution: Ensure system packages are installed:

```
sudo yum install -y python3-devel yara-devel gcc
```

Issue: GCP resource errors (Pub/Sub, BigQuery)

Solution: These are optional. To enable:

11. Create Pub/Sub topics
12. Create BigQuery dataset and tables
13. Or disable publishing in config files

Issue: Permission denied errors

Solution: Some directories require elevated permissions. Scan accessible directories or run with appropriate permissions.

Issue: Python 3.6 compatibility errors

Solution: Refer to [Python 3.6 Compatibility Fixes](#python-36-compatibility-fixes) section.

Directory Structure

```
~/threat-hunting-test/
├── config/                    # Configuration files
│   ├── thor_config.json
│   ├── asgard_config.json
│   └── valhalla_config.json
├── logs/                      # Log files
├── data/                      # Data storage
│   ├── yara_rules/           # YARA rule files (5 test rules)
│   ├── iocs/                 # IOC files
│   ├── sigma_rules/          # Sigma rule files
│   └── test_malware/         # Test malware files (5 files)
├── scripts/                   # Helper scripts
│   ├── setup_env.sh
│   ├── update_config.sh
│   ├── quick_test.sh
│   └── verify_setup.sh
├── tests/                     # Test files
├── docs/                      # Documentation
├── venv/                      # Python virtual environment
├── thor_endpoint_agent.py
├── asgard_orchestration_agent.py
├── valhalla_feed_manager.py
├── threat_hunting_quickstart.py
├── requirements_threat_hunting.txt
└── THREAT_HUNTING_README.md
```

Configuration Files

thor_config.json

- **gcp_project_id:** chronicle-dev-2be9

- **yara_rules_path:** /home/app/threat-hunting-test/data/yara_rules (updated for testing)
- **pubsub_topic:** thor-findings (requires GCP resource)
- **bigquery_table:** soc_data.thor_scan_results (requires GCP resource)

asgard_config.json

- **gcp_project_id:** chronicle-dev-2be9
- **pubsub_topics:** Various topics for campaign management
- **firestore_collection:** campaigns

valhalla_config.json

- **gcp_project_id:** chronicle-dev-2be9
- **gcs_bucket:** valhalla-threat-intel (requires GCP resource)
- **firestore_collections:** yara_rules, ioc_feeds

Testing and Validation

Step 1: Register Endpoint with ASGARD

Status: ✅ Completed

The GCP instance was successfully registered as an endpoint for ASGARD campaigns.

Process:

```
cd ~/threat-hunting-test
source venv/bin/activate

# Create registration script
cat > register_endpoint.py << 'ENDPOINTEOF'
#!/usr/bin/env python3
import sys
import os
import socket
from datetime import datetime
from asgard_orchestration_agent import ASGARDOrchestrationAgent, EndpointInfo

def get_local_endpoint_info():
    hostname = socket.gethostname()
    try:
        s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        s.connect(("8.8.8.8", 80))
        ip_address = s.getsockname()[0]
        s.close()
    except Exception:
        ip_address = "127.0.0.1"

    return {
        "endpoint_id": f"endpoint_{hostname}",
        "hostname": hostname,
        "ip_address": ip_address,
        "os_type": "linux",
        "os_version": "RHEL 8",
        "agent_version": "1.0.0",
```

```
            "last_seen": datetime.utcnow().isoformat(),
            "labels": {
                "env": "production",
                "role": "threat_hunting_test",
                "zone": "asia-southeast2-a",
                "project": "chronicle-dev-2be9"
            },
            "groups": ["production", "threat_hunting"],
            "status": "online",
            "capabilities": ["yara", "ioc", "process", "network", "filesystem"]
        }

    def main():
        print("=" * 60)
        print("ASGARD Endpoint Registration")
        print("=" * 60)

        asgard = ASGARDOrchestrationAgent()
        endpoint_data = get_local_endpoint_info()
        endpoint = EndpointInfo(**endpoint_data)
        success = asgard.register_endpoint(endpoint)

        if success:
            print(f"✓ Successfully registered endpoint: {endpoint.hostname}")
            print(f"  Endpoint ID: {endpoint.endpoint_id}")
            print(f"Total registered endpoints:
    {len(asgard.registered_endpoints)}")
        return 0

    if __name__ == "__main__":
        sys.exit(main())
    ENDPOINTEOF

    chmod +x register_endpoint.py
    python register_endpoint.py
```

Results:

- ✅ Endpoint registered: `xdgaisocapp01`
- ✅ Endpoint ID: `endpoint_xdgaisocapp01`
- ✅ IP Address: `10.45.254.19`
- ✅ Total registered endpoints: 1

Step 2: Create Custom YARA Rules

Status: ✅ Completed

YARA Rules Created:

14. **Test_Ransomware_Signature** - Detects ransomware keywords (encrypt, ransom, decrypt)
15. **Test_Backdoor_Detection** - Detects backdoor patterns (cmd.exe, powershell, backdoor)
16. **Test_Webshell_Detection** - Detects webshell signatures (eval, base64_decode, system, exec)
17. **Test_CryptoMiner_Detection** - Detects cryptocurrency miners (mining, bitcoin, mining_pool)
18. **Test_Suspicious_Process** - Detects suspicious processes

Configuration Update:

```
# Updated config/thor_config.json to use local YARA rules path
yara_rules_path: /home/app/threat-hunting-test/data/yara_rules
```

Step 3: Create Test Malware Files

Status: ✅ Completed

Test Files Created:

19. `test_ransomware.txt` - Triggers ransomware rule
20. `test_backdoor.txt` - Triggers backdoor rule
21. `test_webshell.txt` - Triggers webshell rule
22. `test_cryptominer.txt` - Triggers cryptominer rule
23. `test_suspicious.txt` - Triggers suspicious process rule

Step 4: Test Threat Detection

Status: ✅ Completed - **6 Threats Detected**

Test Command:

```
python thor_endpoint_agent.py --config config/thor_config.json --scan-type
filesystem --target data/test_malware --load-yara
```

Test Results:

Test_Ransomware_Signature	HIGH	test_ransomware.txt	85%
Test_Backdoor_Detection	HIGH	test_backdoor.txt	85%
Test_Webshell_Detection	HIGH	test_webshell.txt	85%
Test_CryptoMiner_Detection	MEDIUM	test_cryptominer.txt	85%
Test_Suspicious_Process	MEDIUM	test_suspicious.txt	85%

Summary:

- Files scanned: 5
- Threats detected: 6
- HIGH severity: 3
- MEDIUM severity: 3
- Detection rate: 100% (all test files detected)

Additional Fix Applied:

- Fixed YARA StringMatch handling for Python 3.6 compatibility
- Replaced subscript access with attribute access for YARA matches

---

Next Steps

1. Create GCP Resources (Optional)

Pub/Sub Topics:

```
gcloud pubsub topics create thor-findings --project=chronicle-dev-2be9
gcloud pubsub topics create thor-scan-requests --project=chronicle-dev-2be9
gcloud pubsub topics create asgard-campaigns --project=chronicle-dev-2be9
```

BigQuery Dataset and Tables:

```
bq mk --dataset chronicle-dev-2be9:soc_data
bq mk --table chronicle-dev-2be9:soc_data.thor_scan_results \

scan_id:STRING,hostname:STRING,start_time:TIMESTAMP,end_time:TIMESTAMP,threats_
detected:INTEGER
```

GCS Bucket:

```
gsutil mb gs://valhalla-threat-intel
```

2. Set Up Threat Intelligence Feeds

24. Obtain API keys for ThreatFox and MalwareBazaar
25. Update VALHALLA configuration with API keys
26. Test feed updates

3. Integration Testing

27. Test THOR → TAA integration (via Pub/Sub)
28. Test ASGARD → THOR workflow
29. Test VALHALLA → THOR rule distribution

Verification Checklist

- [x] Directory structure created
- [x] All files downloaded
- [x] System packages installed
- [x] Python virtual environment created
- [x] All Python packages installed
- [x] Python 3.6 compatibility fixes applied
- [x] Configuration files updated
- [x] VALHALLA initialized successfully
- [x] THOR scanning working
- [x] ASGARD campaign creation working
- [x] **Endpoint registered with ASGARD**
- [x] **Custom YARA rules created and tested**
- [x] **Test malware files created**
- [x] **Threat detection verified (6 threats detected)**
- [ ] GCP resources created (optional)

Support and Maintenance

Logs Location

- Application logs: `logs/`
- GCP service logs: Check Cloud Logging in GCP Console

Updating Components

30. Pull latest code from GitHub
31. Reapply Python 3.6 compatibility fixes if needed
32. Test components individually
33. Update configuration if needed

Backup Recommendations

- Backup configuration files
- Backup custom YARA rules
- Backup campaign data (if using Firestore)

Appendix: Quick Reference Commands

Environment Setup

```
cd ~/threat-hunting-test
source venv/bin/activate
```

Run THOR

```
python thor_endpoint_agent.py --config config/thor_config.json --scan-type
filesystem --target /tmp
```

Run VALHALLA

```
python valhalla_feed_manager.py
```

Run ASGARD

```
python asgard_orchestration_agent.py --help
```

Verify Setup

```
bash scripts/quick_test.sh
bash scripts/verify_setup.sh
```

Testing Results Summary

Endpoint Registration

- **Status:** ✅ Success
- **Endpoint:** xdgaisocapp01
- **Endpoint ID:** endpoint_xdgaisocapp01

- **IP Address:** 10.45.254.19
- **Registered Endpoints:** 1

YARA Rules Testing

- **Rules Created:** 5 test rules
- **Rules Location:** `~/threat-hunting-test/data/yara_rules/test_malware_rules.yar`
- **Rules Loaded:** 2 YARA rule files successfully loaded

Threat Detection Test

- **Test Files:** 5 test malware files created
- **Files Scanned:** 5
- **Threats Detected:** 6
- **Detection Rate:** 100%
- **HIGH Severity:** 3 threats
- **MEDIUM Severity:** 3 threats

Test Execution Details

- **Scan ID:** thor_scan_20260105_081410
- **Hostname:** xdgaisocapp01
- **Duration:** ~0.37 seconds
- **All YARA rules triggered successfully**

Document History

1.1	2026-01-05	Deployment Team	Added endpoint registration and testing results
1.0	2026-01-05	Deployment Team	Initial deployment documentation

---

End of Document