SYHUNT

Syhunt Documentation

User and Integration Guide

Version 7.2

Documentation for **Syhunt Hybrid** and its modules: Syhunt Dynamic (DAST), Syhunt Code and Mobile (SAST/MAST), Syhunt API and Syhunt Forensic.

New to Syhunt? Start with the Welcome Guide, then the Quick Start.

Generated on 2026-10-26 · www.syhunt.com

Syhunt Hybrid 7.2

Contents

GETTING STARTED

Welcome to Syhunt
Quick Start
Download
Proof-of-Concept Test Plan
FAQ

INSTALLATION

Installation Overview and Windows
Installing on Linux
Installing on macOS
Web UI
Installing Wine
Migration Notes
Performance Tuning

SCANNING

Dynamic Scanning (DAST)
Authenticated Scanning
Source Code Scanning (SAST/MAST)
Web API Scanning
Hunt Methods
Ignore Rules
Scheduling Scans
Forensic Log Analysis
Reporting

INTEGRATIONS

Command Line (CLI)
PowerShell
Web API
Lua API
GitHub
GitLab
Bitbucket
Azure DevOps
Jenkins
Issue Trackers
DefectDojo
Faraday

REFERENCE

Technical Specifications
Preferences
Vulnerability Checks

Page 2 of 175

Syhunt Hybrid 7.2

SUPPORT

Reporting Bugs and Getting Support

Page 3 of 175

Syhunt Hybrid 7.2

Getting started

Page 4 of 175

Syhunt Hybrid 7.2

Welcome to Syhunt

Thank you for choosing Syhunt. Our goal is to help you get the most out of the Syhunt tools so your organization can reach stronger application security and compliance faster, by finding and fixing vulnerabilities and weaknesses before they reach production.

Start with the four steps below, then read the rest of this guide.

Initial steps

1. Activate your license

If you haven't activated your license yet, do it now so you can start using the Syhunt tools.

1. Download the setup executable from the link provided by Syhunt or your Syhunt Authorized Partner.
2. Double-click the downloaded file and follow the installer (a standard *Next* → *Next* → *Finish* process). Syhunt launches when you click **Finish**.
3. Enter your Product Key File, either by following the instructions in the email, or by enabling the Web UI, which prompts you for it. Use the key file attached to the email that contained your download link.
4. Click **Apply**. A success message confirms that Syhunt is ready, and the available options become enabled on the Home screen.

If Syhunt doesn't ask for a key, a license is already active and must be replaced. To replace it, open `https://127.0.0.1:8018/syhunt/?n=license` or run again the CLI command you used to activate your key.

2. Subscribe to notifications

Subscribe to the Customer Newsletter to receive an email whenever product updates and enhancements become available. Updates can be downloaded and installed directly from the Syhunt website. Syhunt 6.8.4 and later also announce new releases in the Launcher and in reports.

- Subscribe here

3. Follow the POC test plan

We strongly recommend working through the Syhunt Proof-of-Concept Test Plan. It walks you through scans under a variety of conditions that simulate real-world use. By the end, you will be familiar with the key product features and integration options.

- Test plan (Markdown)

Page 5 of 175

Syhunt Hybrid 7.2

- Test plan (PDF)

4. Read the quick start guides

Set aside time for you and your technical staff to read the quick start guides and the other documents in this repository, and get familiar with the tools and features your organization plans to use.

- Documentation index
- Syhunt Support area

Getting help

The Syhunt Support area is the fastest way to find answers. If you have a question or run into a problem, open a support ticket. When reporting a false positive or any other issue, follow the bug reporting guidelines for a faster resolution.

Requesting features

You can help shape the product roadmap. Send feature and improvement requests to our product management team at any time by opening a support ticket.

Viewing and renewing your license

Syhunt or your Syhunt Authorized Partner will contact you about 90 days before your license ends so you can renew your license or support coverage. Syhunt 6.7.1 and later also show a reminder in the main interface about 60 days before the end date. For questions about this process or about additional requirements, contact Syhunt or your Syhunt Authorized Partner.

You can check your license term and details at any time from **Help** → **License**.

Training

Training may be available in your region and language. Contact your Syhunt Authorized Partner to check availability, fees and schedules.

That's all for now. For anything else, feel free to contact us.

Page 6 of 175

Syhunt Hybrid 7.2

Quick Start

> **Applies to:** Syhunt Hybrid 7.2

Syhunt Hybrid is a multi-language web application security assessment suite. It scans for the most common web application flaws from an attacker's perspective: it injects data into web applications and analyzes their responses to determine whether the code is vulnerable to specific attacks, such as SQL injection, cross-site scripting (XSS) and many others. When requested, Syhunt also scans the application's source code for security issues.

What can Syhunt scan?

Syhunt is designed to cover a wide range of platforms:

- **Any web server platform** (through dynamic scanning), whether it runs on Unix, Linux or Windows.
- **Devices that serve web pages**, such as routers and firewalls.
- **Web applications written in** ASP.NET, Java, Node.js, Lua, Perl, PHP, Python, Ruby and many other languages (through source code scanning).
- **Web application firewalls and intrusion detection systems** (through its evasion techniques).

A proactive approach

Web application scanners are often seen as being like antivirus software: reactive, and dependent on frequent signature updates, daily or at least weekly. That is true of traditional scanners. Since 2008, however, Syhunt has taken a proactive approach to building its checks and its hybrid scanner, focusing on common weaknesses (CWEs) rather than on individually disclosed vulnerabilities (CVEs).

Traditional, reactive scanners	Syhunt's proactive approach
Depend heavily on regular signature updates that target CVEs (**known, disclosed vulnerabilities** in third-party applications).	Does not depend on frequent updates. Development focuses on a list of CWEs (**common weaknesses**). CVE-specific checks ship with minor and major releases as a **complement** to the proactive checks.
Cannot detect undisclosed vulnerabilities in third-party applications.	Detects **both disclosed and undisclosed vulnerabilities** in third-party applications: accurately detecting a single CWE means matching hundreds of CVEs at once.
Miss most, if not all, vulnerabilities in your custom applications.	Is much better suited to finding vulnerabilities in your custom applications.
Cannot analyze source code.	Performs **hybrid analysis**: dynamic analysis complements source code analysis (DAST + SAST).

Next steps

Task	Guide
Run your first dynamic (black-box) scan	Dynamic Scanning
Run your first source code (white-box) scan	Source Code Scanning
Choose the right scan method	Hunt Methods
Schedule recurring scans	Scheduling Scans
Customize reports	Reporting
Check what you need before a dynamic scan	Pre-scan requirements

Page 7 of 175

Syhunt Hybrid 7.2

Task	Guide
Check hardware and software requirements	Technical Specifications

Redefining the severity of a vulnerability

If you disagree with the severity assigned to a reported vulnerability, you can set a new severity that applies to all future scans:

1. In the Web UI, open the Connectors screen (☰ **Menu** → **Connectors**).
2. Add a connector of type **List** named `VULNSEVERITY`.
3. Enter the vulnerability's unique check ID, followed by `=` and the new severity. For example: `C-1603660271-7241=low`
4. Click **Add** to add the entry to the list.
5. Click **Save** to save the list.

Page 8 of 175

Syhunt Hybrid 7.2

Download

Syhunt installers are distributed through the Syhunt downloads page.

Edition	How to get it
Syhunt Hybrid and **Syhunt Runner**	Use the download link and customer key sent to you by Syhunt or your Syhunt Authorized Partner.
Demo / trial	Use the trial download link you received after requesting a demo.
Syhunt Community	Available on the downloads page without a key.
Prey Server (vulnerable test applications)	See Prey Server in the POC test plan.

If your license has expired, contact Syhunt to renew it.

Once you have the installer, follow the guide for your platform:

- Windows
- Linux
- macOS
- Wine (Linux and macOS)

Page 9 of 175

Syhunt Hybrid 7.2

Proof-of-Concept Test Plan

> **Applies to:** Syhunt Hybrid 7.2

> **Note**
> This test plan was written for **Syhunt Hybrid**, the full-featured edition. Syhunt Community has restricted functionality and vulnerability checks, so some test cases cannot be performed with it and not every vulnerability will be detected.

Introduction

This test plan verifies the functionality, accuracy and correct operation of every key part of Syhunt Hybrid. It is meant to be carried out on the customer's premises after Syhunt Hybrid has been deployed and activated, and it consists of dynamic and static scans run under various conditions that simulate real-world use. By the end, you will be familiar with the main product features and integration options, and Syhunt will be able to collect your feedback on usability.

Test cases

Phase	Test case
1. Dynamic auditing	D1: Scan a live website for vulnerabilities
	D2: Log in manually through the browser and scan a restricted website
	D3: Log in automatically and scan a restricted website
	D4: Map a website only (crawling test with JavaScript execution)
	D5: Perform an incremental dynamic scan
	D6: Perform concurrent dynamic scans
2. Source code auditing	S1: Scan local source code files
	S2: Scan remote Git repositories
3. Integration	N1: Launch a code scan from the command line
	N2: Launch a dynamic scan from the command line
	N3: Create a tracker issue from a reported vulnerability
	N4: Generate an F5 BIG-IP ASM–compatible export
4. Reporting	R1: Generate a standard HTML report
	R2: Generate a complete HTML report
	R3: Export scan results as XML
	R4: Edit a reported vulnerability
	R5: Generate a comparison report between scans

Environment requirements

The results in this plan were obtained on Windows 11 x64 with an Intel Core i9, 64 GB of RAM, a 1 TB SSD and Internet access. Make sure you have a fast SSD and Internet access, and that you meet at least the minimum system requirements.

Page 10 of 175

Syhunt Hybrid 7.2

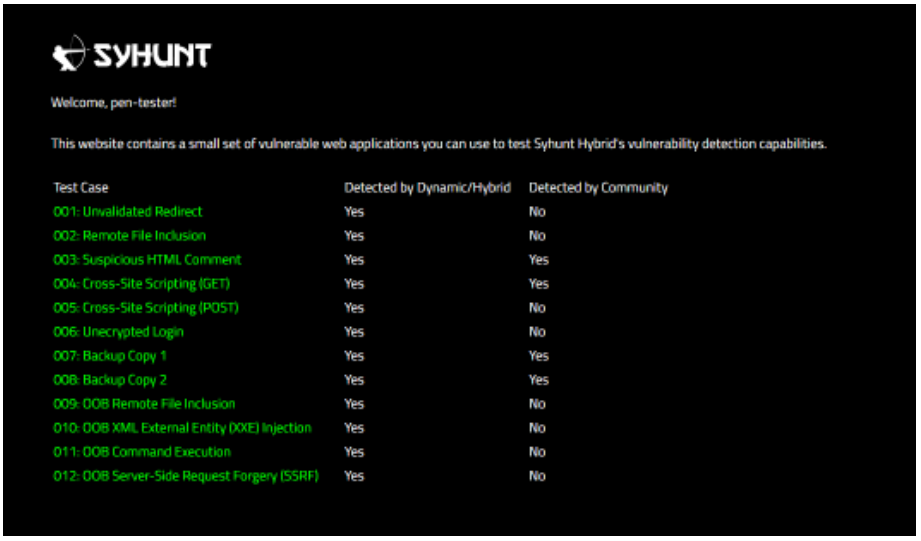
The plan assumes Syhunt Hybrid is already installed on the Windows machine that will run the tests, and that the Web UI is enabled. If it isn't:

1. Download the setup executable from the link provided by Syhunt, double-click it and follow the installer (*Next* → *Next* → *Finish*). Syhunt Hybrid launches when you click **Finish**.
2. Enable the Syhunt Web UI.
3. On the Web UI home page, enter the Pen-Tester Key from the email that contained your download link, and click **Apply**. A success message confirms Syhunt is ready for testing.

Prey Server

Prey Server is a portable Apache/PHP web server bundled with a set of intentionally vulnerable web applications for demonstration purposes.

1. Download it from https://syhunt.fra1.cdn.digitaloceanspaces.com/tools/hybrid_xtras/syhunt-vulnphpserver.zip.
2. Unzip it to a folder of your choice.
3. Run `PreyServer.exe`.
4. Run `mysql\start.exe` to start the MySQL server.
5. Open http://127.0.0.1/syhunt/vulndemo in your browser. You should see the welcome page:



Git for Windows

The Git-related test cases require Git for Windows. Install it with the default settings, then confirm that the `git` command works from the Command Prompt.

Estimated scan times

- **Dynamic scans (phase 1)** take about **5 minutes** each against the recommended targets. The incremental scan in D5 takes about half that.
- **Code scans (phase 2)** take about **10 seconds** against the recommended targets. For comparison, a larger Java + JavaScript project of about 200,000 lines takes around 4 minutes, and an incremental rescan of the same codebase takes about half that. In other words, Syhunt analyzes roughly 6 million lines of code per hour, and about 12 million per hour on recurring scans of the same targets.

Phase 1: Dynamic auditing

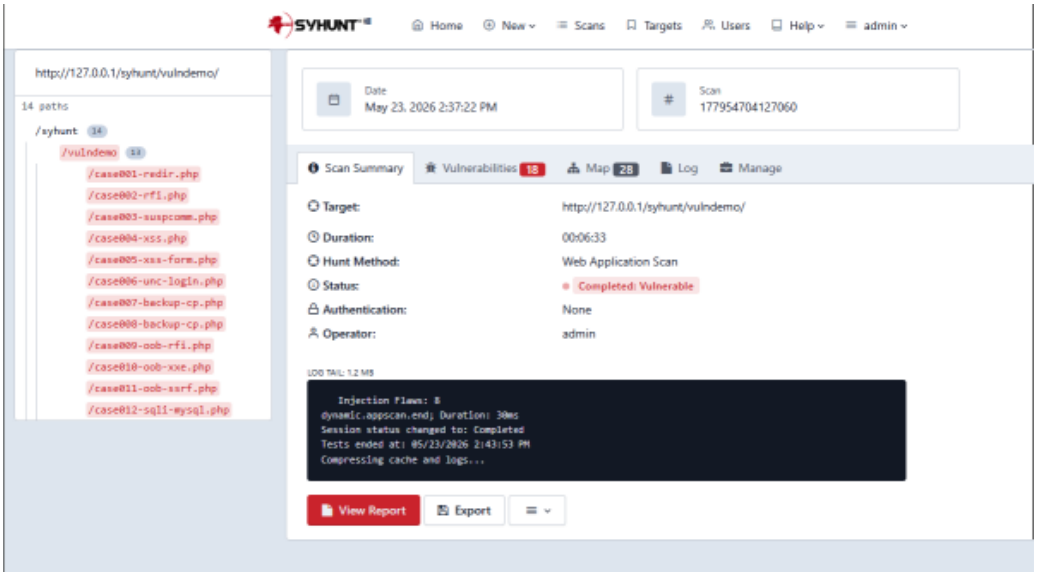
D1. Scan a live website for vulnerabilities

Action: Scan http://127.0.0.1/syhunt/vulndemo for vulnerabilities.

Page 11 of 175

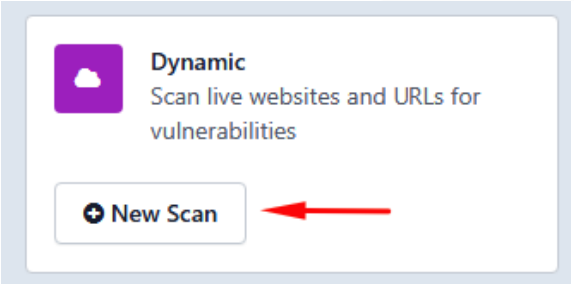
Syhunt Hybrid 7.2

Expected result: When the scan completes, the **Vulnerabilities** tab lists every test case, from 001 to 012.

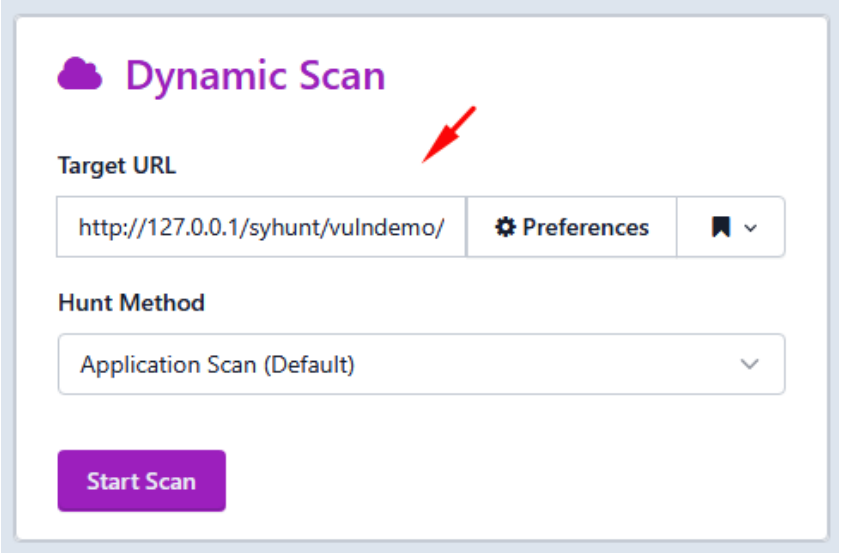


Steps:

1. Make sure Prey Server is running (see Prey Server).
2. Open the Syhunt Hybrid home page, `https://127.0.0.1:8018/syhunt/`, and click Dynamic's **New Scan** button.



3. Enter `http://127.0.0.1/syhunt/vulndemo` as the target URL.



4. Select the **Application Scan (Default)** hunt method, which checks for all vulnerabilities using the recommended settings.
5. Click **Start Scan**.

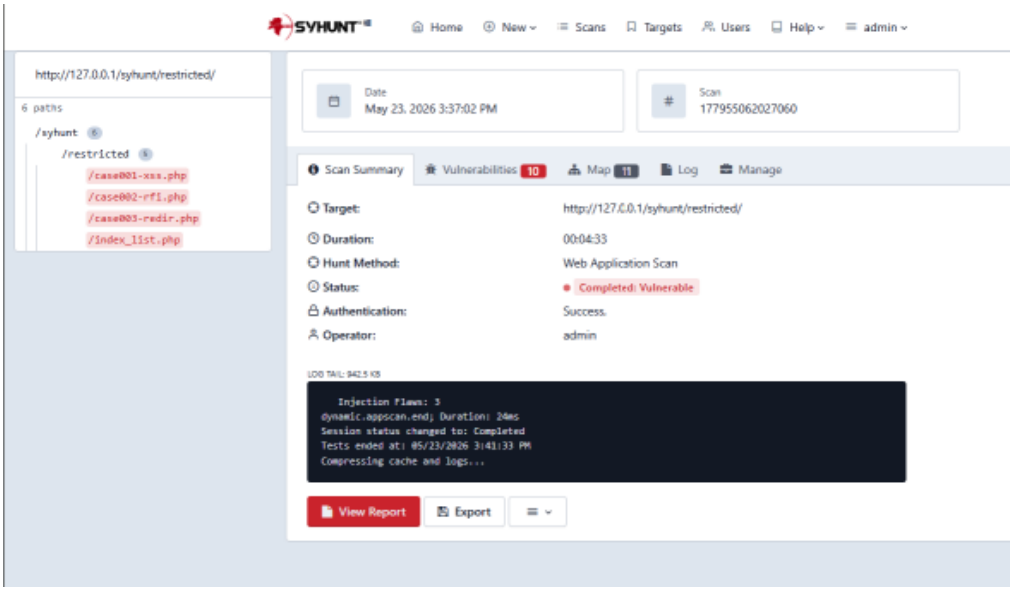
D2. Log in manually through the browser and scan a restricted website

Action: Scan http://127.0.0.1/syhunt/restricted, which uses web form authentication.

Expected result: When the scan completes, the **Vulnerabilities** tab lists every test case, from 001 to 003.

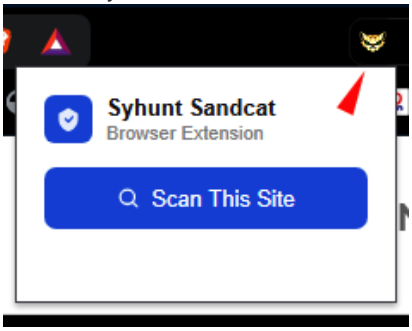
Page 12 of 175

Syhunt Hybrid 7.2



Steps:

1. Make sure Prey Server is running (see Prey Server).
2. Make sure the Syhunt Sandcat browser extension is enabled.
3. In your browser, go to http://127.0.0.1/syhunt/restricted.
4. Log in with username **test** and password **CUBPzjVy**.
5. Click the Syhunt Sandcat icon in the browser toolbar, then **Scan This Site**.



6. A screen with a screenshot of the website and a green **Session Ready** message appears.
7. Select the **Application Scan (Default)** hunt method.
8. Click **Start Scan**.

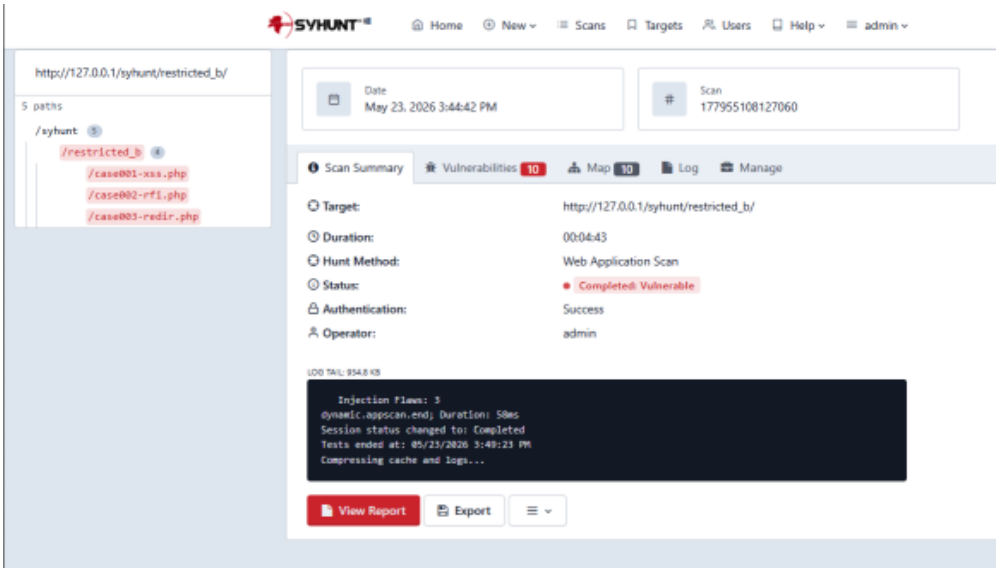
D3. Log in automatically and scan a restricted website

Action: Scan http://127.0.0.1/syhunt/restricted_b, which uses HTTP Basic authentication.

Expected result: When the scan completes, the results list every test case, from 001 to 003.

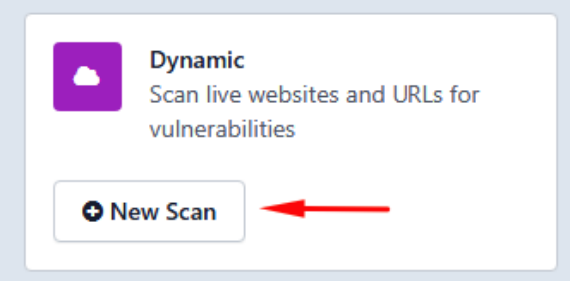
Page 13 of 175

Syhunt Hybrid 7.2

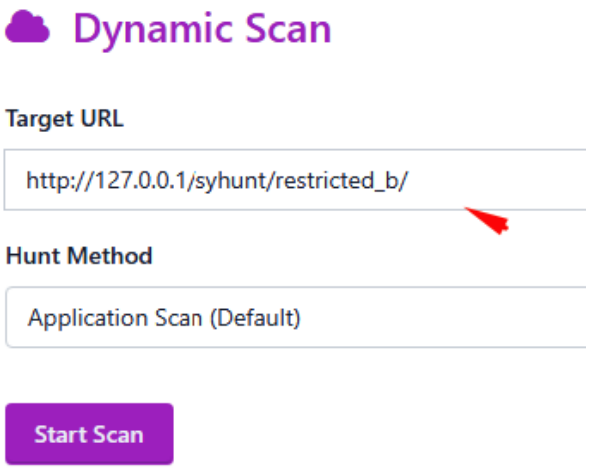


Steps:

1. Make sure Prey Server is running (see Prey Server).
2. Open `https://127.0.0.1:8018/syhunt/` and click Dynamic's **New Scan** button.



3. Enter `http://127.0.0.1/syhunt/restricted_b` as the target URL.

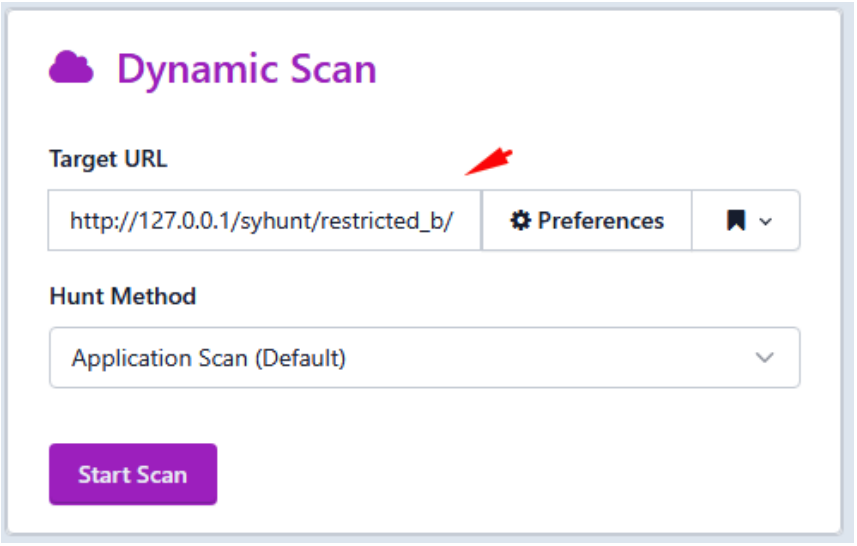


4. Select the **Application Scan (Default)** hunt method.

Page 14 of 175

Syhunt Hybrid 7.2

5. Click **Preferences**.

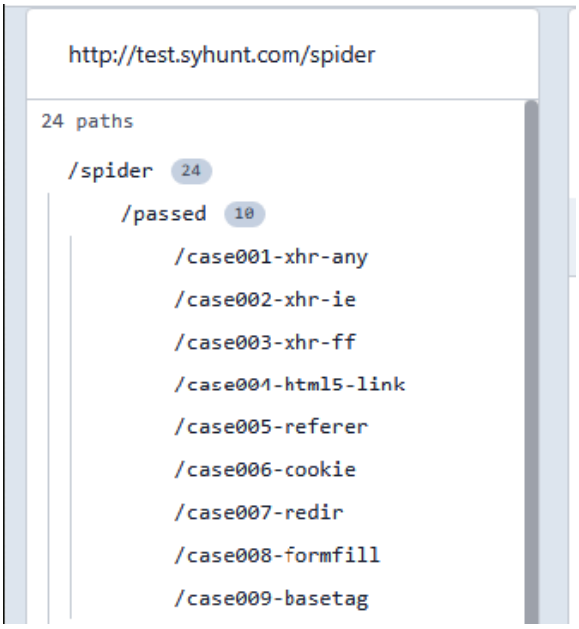


6. On the **Authentication** tab, set **Authentication** to **Basic**.
7. Enter username **test** and password **test**, then click **Save**.
8. Click **Start Scan**.

D4. Map a website only (crawling test with JavaScript execution)

Action: Crawl http://test.syhunt.com/spider.

Expected result: When the scan completes, expanding the site tree in the left sidebar shows the site structure, including every test case from 001 to 009 under the `/passed/` directory.

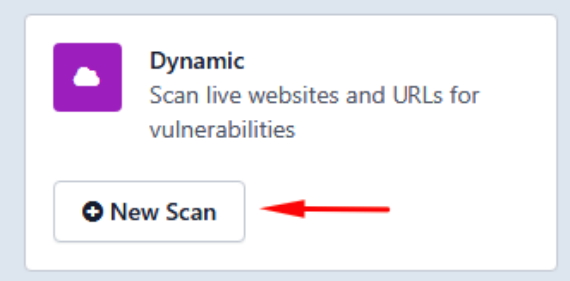


Steps:

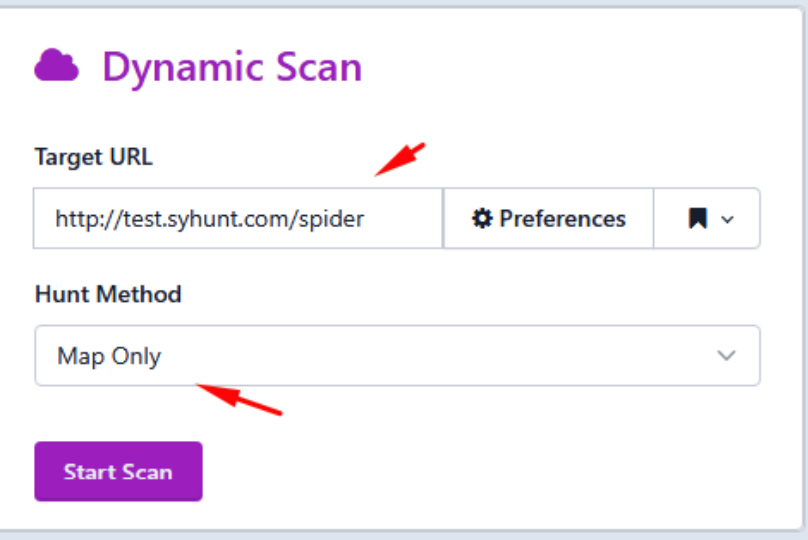
Page 15 of 175

Syhunt Hybrid 7.2

1. Open `https://127.0.0.1:8018/syhunt/` and click Dynamic's **New Scan** button.



2. Enter `http://test.syhunt.com/spider` as the target URL.

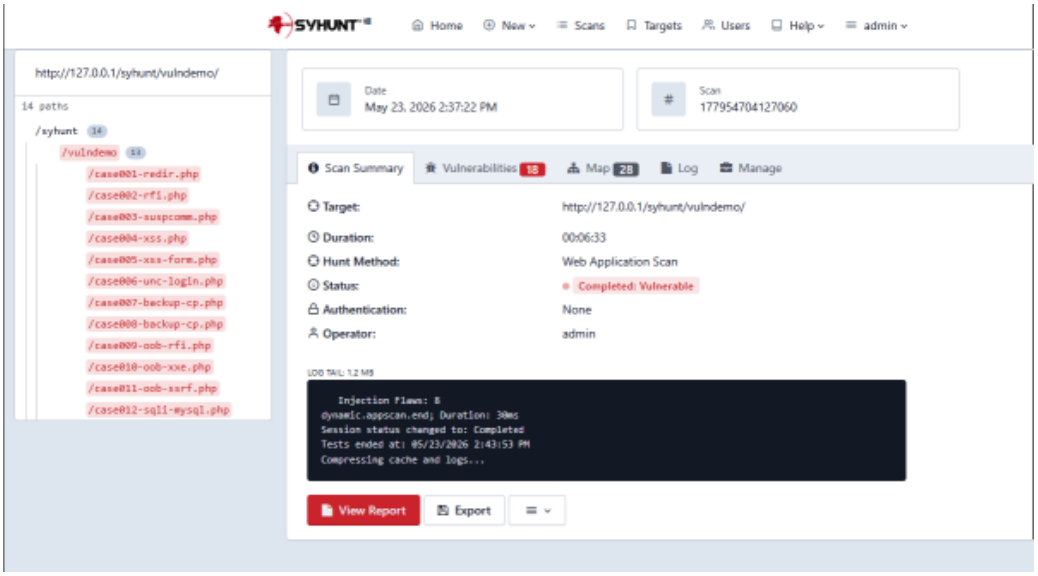


3. Select the **Spider Only** hunt method.
4. Click **Start Scan**.

D5. Perform an incremental dynamic scan

Action: Scan http://127.0.0.2/syhunt/vulndemo twice.

Expected result: The first scan takes about 4 minutes; the second takes about half that (around 2 minutes). After each scan, the results list every test case, from 001 to 012.



Steps:

1. Follow the steps in D1, but use `127.0.0.2` instead of `127.0.0.1` as the target, since incremental scanning is not enabled for `127.0.0.1`.

Page 16 of 175

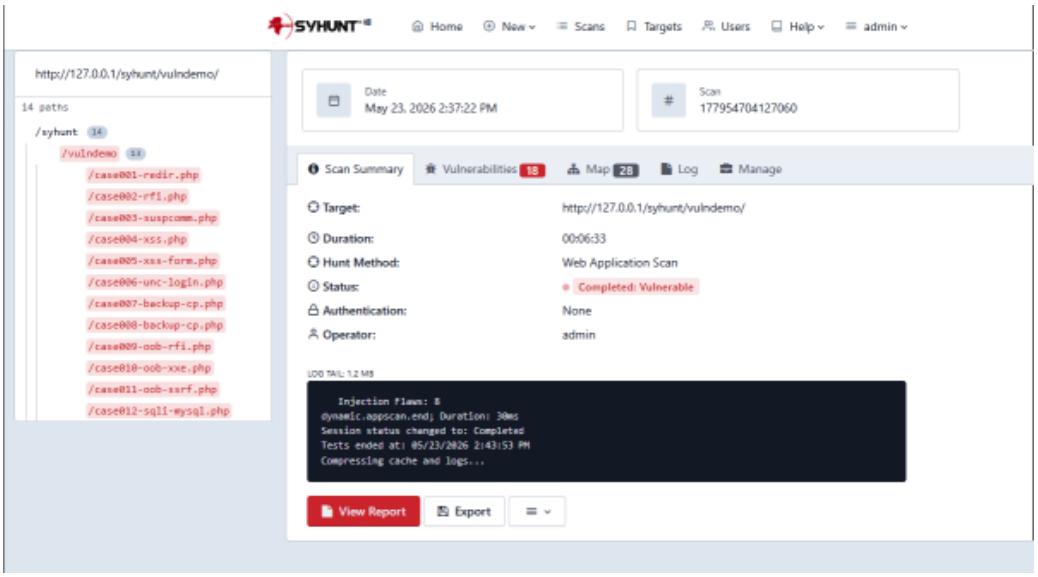
Syhunt Hybrid 7.2

2. Repeat the same steps to scan `127.0.0.2` a second time.

D6. Perform concurrent dynamic scans

Action: Run two simultaneous scans against http://127.0.0.1/syhunt/vulndemo.

Expected result: When the scans complete, the **Vulnerabilities** tab of each scan lists every test case, from 001 to 012. The right sidebar shows the status of each scan, marked in red.



Steps:

1. Follow the steps in D1.
2. Before the scan ends, return to the Syhunt Hybrid home screen and repeat steps 2 to 5 of D1 to start a second scan against the same target.

Phase 2: Source code auditing

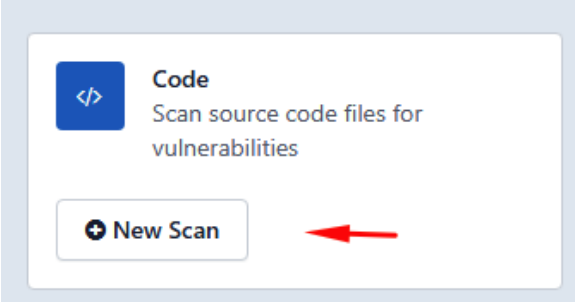
S1. Scan local source code files

Action: Launch a code scan against a local archive from the user interface.

Expected result: When the scan completes, the **Vulnerabilities** tab lists every test case, from 001 to 009.

Steps:

1. Download https://github.com/syhunt/vulnphp/archive/master.zip with your browser.
2. Open the Syhunt Hybrid home page and click Code's **New Scan** button.



3. On the **File** tab, click **Choose File** and select the `master.zip` file you downloaded.
4. Make sure the **Code Scan (Default)** hunt method is selected.
5. Click **Start Scan**.

You can repeat this test with vulnerable samples in other languages (the number of test cases varies by archive):

Page 17 of 175

Language	Archive
Java	https://github.com/syhunt/vulnjava-wavsep/archive/master.zip
Lua	https://github.com/syhunt/vulnlua/archive/master.zip
PHP	https://github.com/syhunt/vulnphp/archive/master.zip

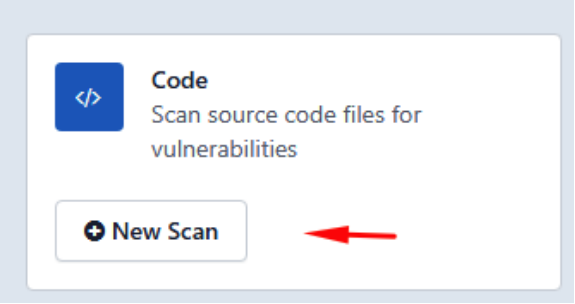
S2. Scan remote Git repositories

Action: Launch a code scan against a remote Git repository.

Expected result: When the scan completes, the results list every test case, from 001 to 009.

Steps:

1. Open the Syhunt Hybrid home page and click Code's **New Scan** button.



2. Enter `https://github.com/syhunt/vulnphp.git` as the project URL.
3. Make sure the **Code Scan (Default)** hunt method is selected.
4. Click **Start Scan**.

You can repeat this test with repositories in other languages (the number of test cases varies by repository):

Language	Repository
Java	`https://github.com/syhunt/vulnjava-wavsep.git`
Lua	`https://github.com/syhunt/vulnlua.git`
PHP	`https://github.com/syhunt/vulnphp.git`

Phase 3: Integration

N1. Launch a code scan from the command line

Action: Scan https://github.com/syhunt/vulnphp from the command-line interface.

Expected result: When the scan completes, the results list every test case, from 001 to 009.

Steps:

1. Open a Command Prompt in the Syhunt Hybrid installation folder (see Command Line).
2. Run:

```
scancode https://github.com/syhunt/vulnphp.git
```

N2. Launch a dynamic scan from the command line

Action: Scan http://127.0.0.1/syhunt/vulndemo from the command-line interface.

Syhunt Hybrid 7.2

Expected result: When the scan completes, the results list every test case, from 001 to 012.

Steps:

1. Make sure Prey Server is running (see Prey Server).
2. Open a Command Prompt in the Syhunt Hybrid installation folder (see Command Line).
3. Run:

```
scanurl 127.0.0.1/syhunt/vulndemo
```

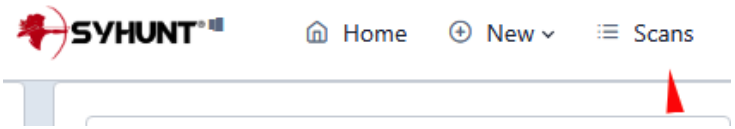
N3. Create a tracker issue from a reported vulnerability

Action: Create a GitHub issue from a reported vulnerability.

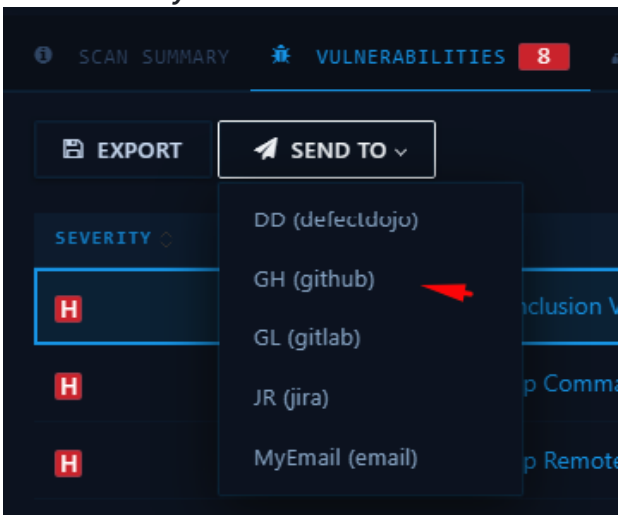
Expected result: After you add a GitHub connector and submit a vulnerability, the vulnerability appears as an issue in your GitHub repository.

Steps:

1. Create a GitHub account if you don't have one, and create a test repository.
2. Add a GitHub connector in Syhunt as described in Issue Trackers.
3. Open **Scans** from the menu.



4. Click a scan with status **Vulnerable** and switch to the **Vulnerabilities** tab.
5. Select the vulnerability you want to send to GitHub.
6. Click **Send To** → ***your connector name*** to submit it.



N4. Generate an F5 BIG-IP ASM–compatible export

Action: Generate an ASM export file for a scan.

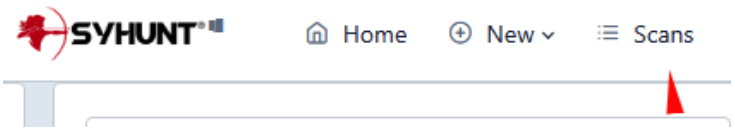
Expected result: After you click **Download**, the browser opens the compatible XML file, which you can then import into BIG-IP ASM.

Steps:

Page 19 of 175

Syhunt Hybrid 7.2

1. Open **Scans** from the menu.



2. Right-click a scan with status **Vulnerable** and choose **Generate Report**.
3. Select the **Complete** report template.
4. Set **Format** to **XML ASM generic scanner format**.
5. Click **Generate**, then **Download**.

Phase 4: Reporting

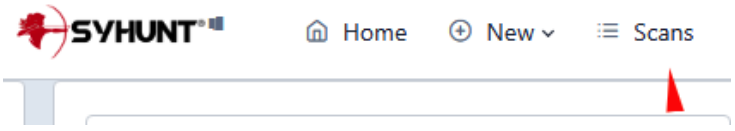
R1. Generate a standard HTML report

Action: Generate a report for a scan, sorting vulnerabilities by CVSS 3.

Expected result: After you click **Generate**, the browser lets you download the HTML report.

Steps:

1. Open **Scans** from the menu.



2. Click a scan with status **Vulnerable**, switch to the **Vulnerabilities** tab and click **Export**.
3. Select the **Standard** report template, if it isn't already selected.
4. Select the **CVSS** sorting method, if it isn't already selected.
5. Click **Generate** (bottom-left corner of the tab), then **Download**.

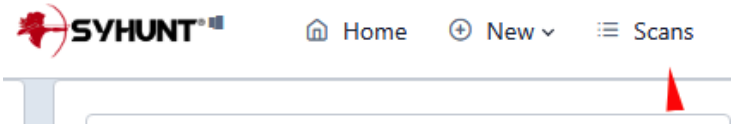
R2. Generate a complete HTML report

Action: Generate a complete report for a scan, sorting vulnerabilities by CVSS 3.

Expected result: After you click **Generate**, the browser lets you download the HTML report.

Steps:

1. Open **Scans** from the menu.



2. Click a scan with status **Vulnerable**, switch to the **Vulnerabilities** tab and click **Export**.
3. Select the **Complete** report template.
4. Select the **CVSS** sorting method, if it isn't already selected.
5. Click **Generate** (bottom-left corner of the tab), then **Download**.

Page 20 of 175

Syhunt Hybrid 7.2

R3. Export scan results as XML

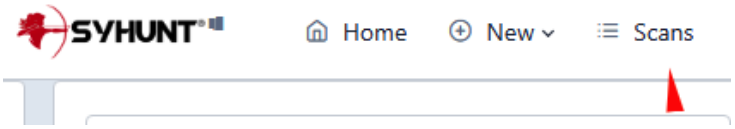
Action: Generate an XML results file for a scan.

Expected result: After you click **Download**, the browser lets you download the XML results file.

```
</vulnerable_code>
▼<cvss>
  ▼<cvss3>
     <vector>AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H</vector>
     <score>9.8 CRITICAL</score>
     <base_score>9.8</base_score>
     <severity>critical</severity>
     <impact_score>5.9</impact_score>
     <exploitability_score>3.9</exploitability_score>
     <temporal_score>NA</temporal_score>
     <environmental_score>NA</environmental_score>
     <mod_impact_subscore>NA</mod_impact_subscore>
   </cvss3>
  ▼<cvss2>
     <vector>AV:N/AC:L/Au:N/C:C/I:C/A:C</vector>
     <score>10.0 HIGH</score>
     <base_score>10.0</base_score>
     <severity>high</severity>
     <impact_score>10.0</impact_score>
     <exploitability_score>10.0</exploitability_score>
```

Steps:

1. Open **Scans** from the menu.



2. Click a scan with status **Vulnerable**, switch to the **Vulnerabilities** tab and click **Export**.
3. Select the **Complete** report template.
4. Set the output format to **XML**.
5. Click **Generate**, then **Download**.

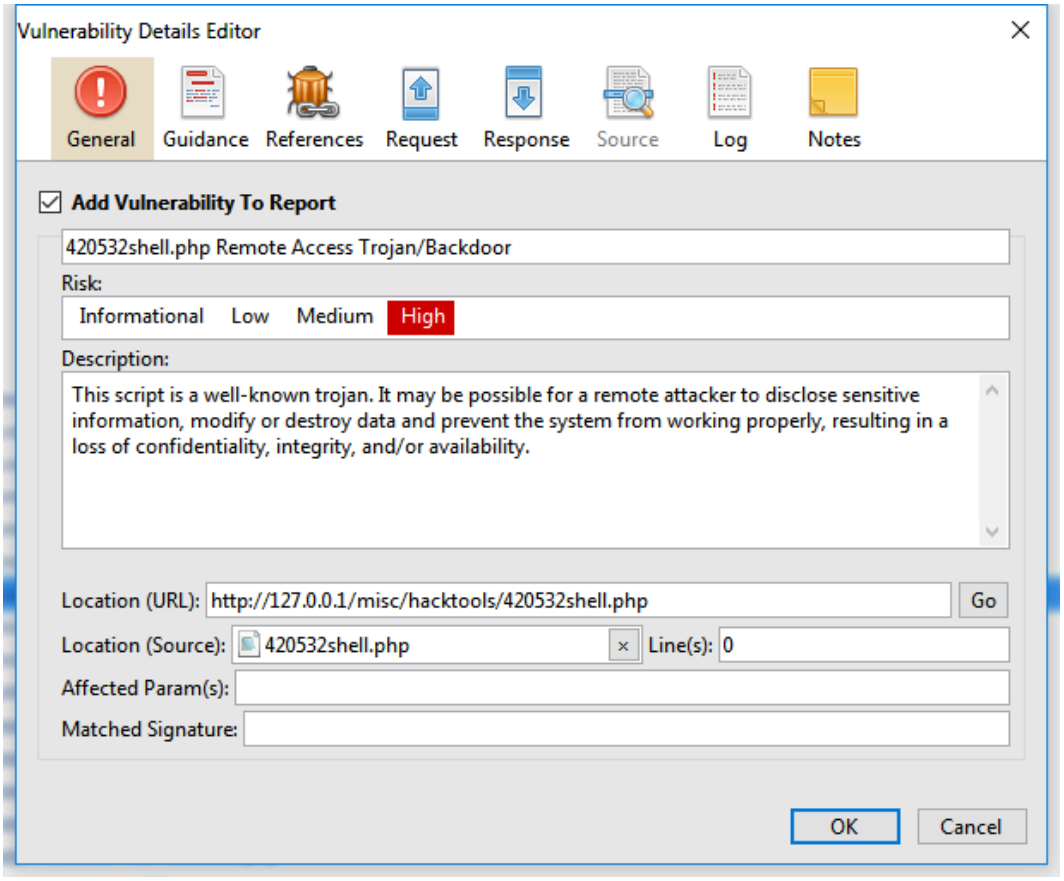
R4. Edit a reported vulnerability

Action: Edit the details of a reported vulnerability.

Expected result: The vulnerability edit screen opens. After you save your changes, any report you generate includes the edited information.

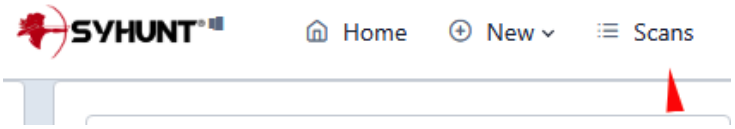
Page 21 of 175

Syhunt Hybrid 7.2



Steps:

1. Open **Scans** from the menu.

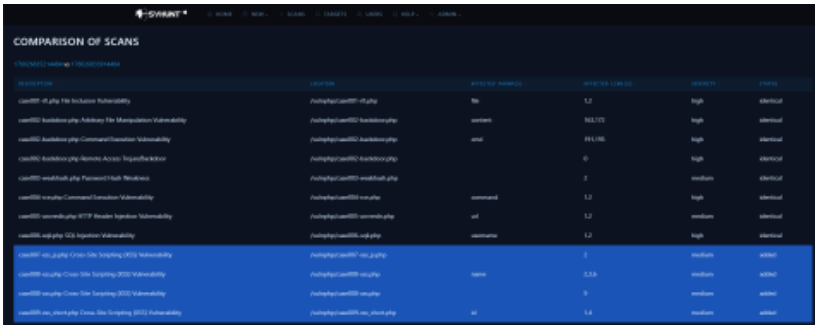


2. Click a scan with status **Vulnerable** and go to the **Vulnerabilities** tab.
3. Click a vulnerability. Its details screen opens.
4. On the **Notes** tab, add some notes.
5. Click **Save**.

R5. Generate a comparison report between scans

Action: Generate a report comparing two scans.

Expected result: The generated report matches the comparison screen and lists test cases 007 to 009 as removed.



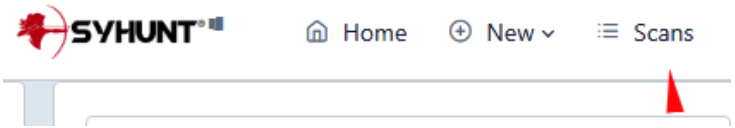
Steps:

1. Follow the steps in S1, but extract the archive to a folder and scan the folder.

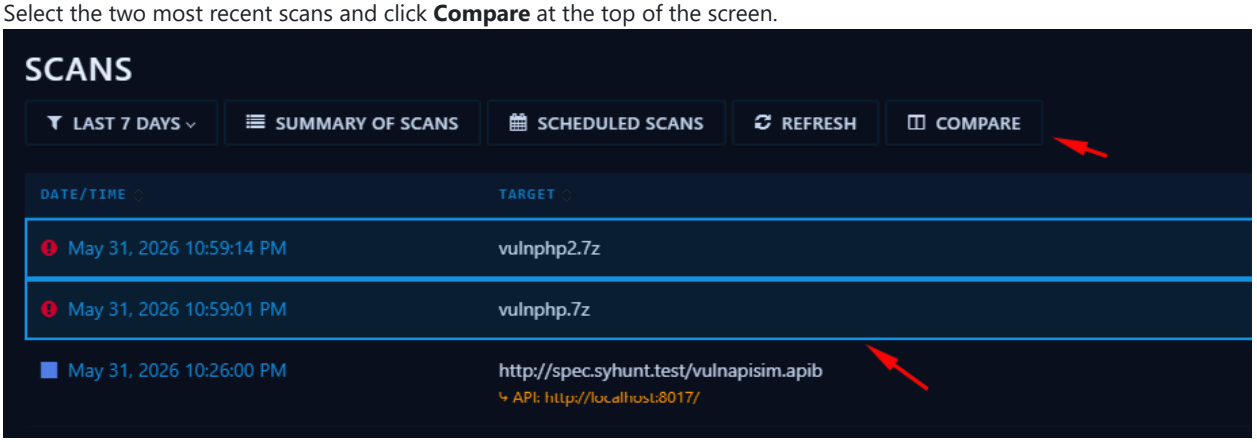
Page 22 of 175

Syhunt Hybrid 7.2

2. When the scan completes, delete test cases 007 to 009 from that folder.
3. Scan the folder again.
4. Open **Scans** from the menu.



5. Select the two most recent scans and click **Compare** at the top of the screen.



Page 23 of 175

Syhunt Hybrid 7.2

FAQ

> **Applies to:** Syhunt Hybrid 7.2

Syhunt Dynamic

How long does Syhunt Dynamic take to run all the tests?

It depends on how many pages and applications the website contains, and on the scan method you choose. After crawling, the web application checks are usually the longest stage, and their duration grows with the size of the site.

Can I reopen a previous scan and generate its reports again?

Yes. Open **Scans** from the menu, click the scan in the list, then click **View Report**, or **Export** to save the data in a specific format.

Is there a list of the tests performed by the current version?

Yes. Open **Help** → **Vulnerability Checks**, or see Vulnerability Checks.

Can any of the tests crash the target host?

Some denial-of-service checks may crash the target. You can disable them by choosing a scan method without DoS tests, such as **Complete Scan, No DoS** (see Hunt Methods).

Does Syhunt Dynamic have problems with personal firewalls?

You'll need to allow Syhunt to make outbound connections. Some software firewalls also cope poorly with heavy traffic, so we don't recommend running a personal firewall on the same machine as Syhunt.

If the scanning machine runs a firewall with outbound filtering, try disabling it. We have occasionally seen firewalls silently block a program's connections without asking the user first.

Can Syhunt scan ports 21 (FTP) or 23 (Telnet)?

No. Syhunt Dynamic is not a general-purpose network scanner; it specializes in web applications and web APIs.

Product names

What happened to Sandcat Platinum, TrustSight and Collapse?

Sandcat Platinum was renamed Sandcat Pro and later became Syhunt Dynamic. The TrustSight and Collapse technologies were merged into Syhunt Dynamic.

Page 24 of 175

Syhunt Hybrid 7.2

Installation

Page 25 of 175

Syhunt Hybrid 7.2

Installation Overview and Windows

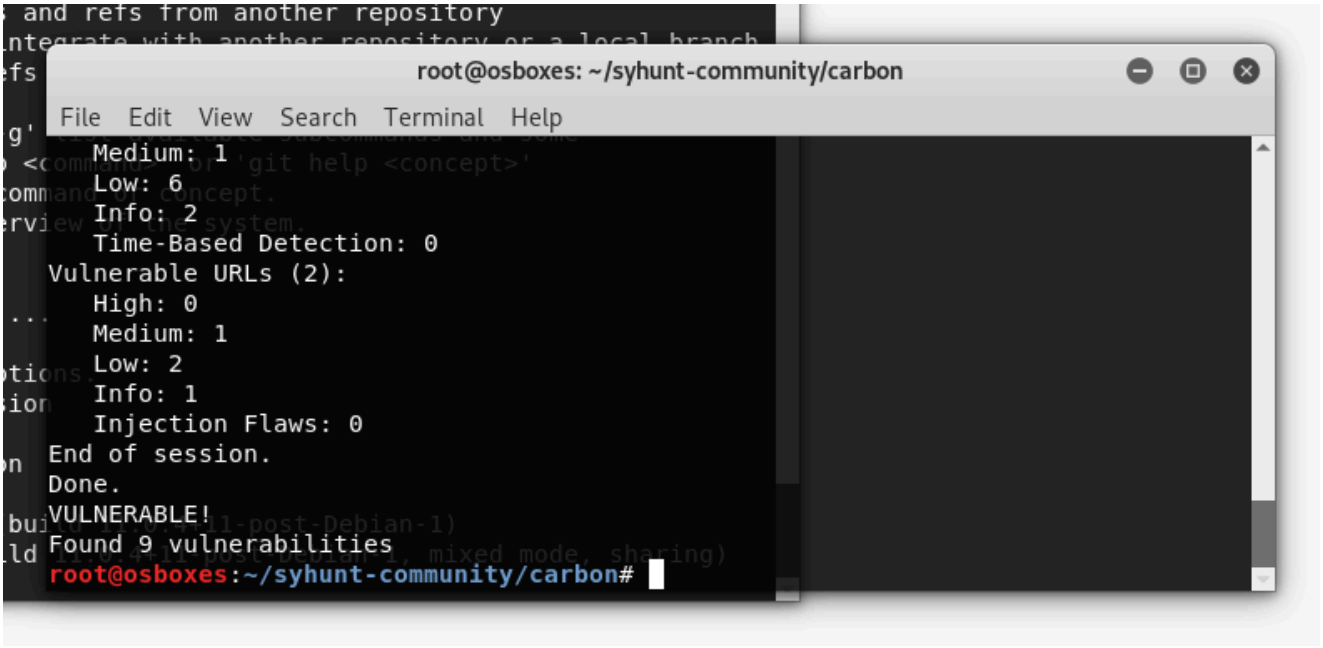
> **Applies to:** Syhunt Hybrid 7.2

Contents

- Introduction
- Windows vs. Linux/macOS editions
- Installing on Windows
- Installing on macOS
- Installing on Linux
- Installing the Syhunt Sandcat browser extension
- System requirements
- Internet connection requirements
- Updating Syhunt
- Uninstalling Syhunt

Introduction

Syhunt Hybrid 7 is a 64-bit, multi-platform application security scanner. It runs on Windows, macOS and Linux distributions such as Ubuntu Desktop/Server, Debian and Fedora with almost no setup effort. Syhunt Hybrid performs web and mobile application security testing (DAST and SAST) and integrates with a variety of systems for continuous integration and scanning.



Windows vs. Linux/macOS editions

	Hybrid for Windows	Hybrid for Linux/macOS
Web UI server	OpenResty for both HTTPS and HTTP	NGINX for HTTPS, OpenResty as back end
Deep AJAX crawler	✅ Chromium engine	✅ Firefox engine
Augmented dynamic analysis (DAST + OAST)	✅	✅
API analysis (AAST + OAST)	✅	✅

Page 26 of 175

Syhunt Hybrid 7.2

	Hybrid for Windows	Hybrid for Linux/macOS
Web application source code analysis (SAST)	✅	✅
Mobile application source code analysis (MAST)	✅	✅
Forensic analysis (FAST)	✅ (CLI only)	✅ (CLI only)
AI-powered capabilities	✅ Patched code examples ✅ AI confidence score	✅ Patched code examples ✅ AI confidence score
Advanced authentication	✅ Recorder, Selenium	✅ Recorder, Selenium
Integrations	✅ Shell (PowerShell) ✅ Issue trackers (GitHub, GitLab, Jira...) ✅ CI/CD tools (GitLab, GitHub, Bitbucket, Jenkins) ✅ WAFs for virtual patching ✅ Azure DevOps / TFS project URLs (non-Git)	✅ Shell (Bash) ✅ Issue trackers (GitHub, GitLab, Jira...) ✅ CI/CD tools (GitLab, GitHub, Bitbucket, Jenkins) ✅ WAFs for virtual patching ✅ Azure DevOps / TFS project URLs (non-Git)
Scan scheduler	✅	✅
Supported operating systems	**Officially:** Windows 11 Windows 10 Windows Server 2016 to 2025	**Officially:** macOS Sequoia 15.4.1 or later (Apple Silicon) Ubuntu Server/Desktop 18.10 and later Debian 9 and later Fedora 32 and later **Unofficially (successfully tested):** Other Linux distributions macOS Monterey
Availability	Available now (native Win64 binaries)	Available now (Wine-powered installation) Ask about native binaries for your distribution

Installing on Windows

Registered customers: follow the initial steps in the Welcome Guide to download, install and start using the full version of Syhunt Hybrid. Non-registered users can use Syhunt Community.

1. *(Optional)* If you plan to scan Git repositories, install Git for Windows.
2. Download the Syhunt Hybrid setup, `syhunt-hybrid-<version>.exe`.
3. Double-click the downloaded file to run it.
4. Read the EULA carefully before accepting it.
5. Complete the installer (*Next* → *Next* → *Finish*). When you click **Finish**, Syhunt launches and the Launcher screen appears.
6. If needed, enable the Syhunt Web UI.

What's next? Read the Quick Start and the integration guides.

Installing on macOS

See the macOS installation guide.

Installing on Linux

Syhunt Hybrid for Linux is the *Carbon* edition: a Wine-powered installation compatible with most 64-bit Linux desktops and servers. See the Linux installation guide for distribution-specific steps.

Page 27 of 175

Installing the Syhunt Sandcat browser extension

Since version 7.2, Syhunt Hybrid includes a browser extension for Edge, Chrome, Brave and other Chromium-based browsers. It enables manual login and browser-assisted testing. The name honors the Sandcat Browser, the built-in browser in Syhunt 7.1 and earlier, which supported manual login in a similar way.

1. In your browser, open **Menu → Extensions → Manage Extensions**.
2. Turn on **Developer mode** (top-right corner).
3. Click **Load unpacked** and select the Sandcat extension folder:
   - Windows: `C:\Program Files\Syhunt Hybrid\Extensions\SandcatExtension`
   - Linux: `/home/<user>/syhunt-hybrid/carbon/sandcat`
   - macOS: `/Applications/Syhunt Hybrid/carbon/sandcat`
4. Syhunt Sandcat now appears in your list of extensions.
5. Open the Web UI home page again. The message **Extension detected and ready** confirms that everything works.

If you reach the Web UI through an IP address, domain or subdomain other than `localhost`, configure its address in the extension:

1. Click the extension's **Details** button, then **Extension options**.
2. Select **Use custom Syhunt UI origin**.
3. Choose the protocol (`http` or `https`).
4. Enter the host, *without* the protocol prefix (`http://` or `https://`).
5. Enter the port used by the Web UI.
6. Click **Save**.
7. Open the Web UI home page again and check for the **Extension detected and ready** message.

System requirements

See Technical Specifications.

Internet connection requirements

The machine running Syhunt must be able to make HTTP(S) requests to the following addresses:

Domain	Ports	Purpose
Your target domain(s)	80, 443*	The domain hosting the web application or codebase you want to scan.
Any or specific asset domains	80, 443	Optional but recommended: lets Syhunt analyze externally hosted JavaScript files and assets during DAST or SAST.
`syhunt.fra1.cdn.digitaloceanspaces.com`	443	Required: installation and auto-update assets.
`www.syhunt.net`	80, 443	Required: assets for HTML/PDF reports, update notifications, auto-update and more.
`signal.syhunt.net`	80, 443	Required during DAST to perform OAST (important).
`api.openai.com`	443	Optional: AI-powered features.
`fonts.googleapis.com`	80, 443	Required: fonts for HTML/PDF reports.
`www.google.com`	443	Required: Google JSAPI for HTML/PDF reports.
`www.gstatic.com`	443	Required: Google JSAPI for HTML/PDF reports.

* If your target uses a non-standard port (such as 8080), or you connect to Git over SSH or another protocol, allow those ports as well.

> **Warning**
> If you use a personal firewall, allow Syhunt to make outbound Internet connections.

On Linux and macOS, the setup is a Java installer. See Running the setup.

Updating Syhunt

Updates can be downloaded and installed from the Syhunt website. Syhunt Hybrid 6.8.4 and later also announce new releases in the Launcher and in reports.

On Linux and macOS, run `scanupdate` to check for updates; if any are available, Syhunt asks whether to download and install them. Run `scanupdate auto` to install available updates without asking for confirmation.

You don't need to uninstall Syhunt before installing a new version, except when updating Syhunt Community CLI on Windows.

Uninstalling Syhunt

Windows: use the uninstall shortcut in the Start menu (under the *Syhunt Hybrid* or *Syhunt Community* folder), or remove Syhunt from **Control Panel** → **Programs and Features**.

Linux and macOS: go to the folder where Syhunt is installed and run:

```
java -jar Uninstall.jar
```

Syhunt Hybrid 7.2

Installing on Linux

> **Applies to:** Syhunt Hybrid 7.2

Syhunt Hybrid runs on Ubuntu (Server or Desktop) and most 64-bit Linux distributions with almost no setup effort, and nearly out of the box on Kali Linux. The Linux edition (*Carbon*) is a Wine-powered installation. For a feature comparison with the Windows edition, see Installation Overview.

Overview

Installing Syhunt on Linux always follows the same pattern:

1. Read and accept the Syhunt EULA before running any of the commands below.
2. Install **Java** (needed by the setup program), if it isn't already installed.
3. *(Optional)* Install **Git** if you plan to scan Git repositories.
4. Install **Wine64**, following the Wine guide for your distribution.
5. Run the Syhunt setup.

The sections below give the exact commands for each distribution.

Distributions: Kali · Parrot OS · Ubuntu · Debian · Fedora · CentOS / RHEL · AlmaLinux · Rocky Linux · Oracle Linux · Amazon Linux 2 · Amazon Linux 2023 · Arch / Manjaro · Linux Mint · MX Linux · openSUSE · elementary OS · Deepin · Solus

Kali Linux

> **Applies to:** Kali Linux 2024.3 or later

Kali already includes most of Syhunt's dependencies, so you only need Wine64 and Syhunt itself.

1. Install Wine64 by following the Debian/Kali Wine guide.
2. Run the Syhunt setup.

Parrot OS

> **Applies to:** Parrot OS 6.1 or later

Like Kali, Parrot OS already includes most of Syhunt's dependencies.

1. Install Wine64 by following the Debian Wine guide.
2. Run the Syhunt setup.

Ubuntu

> **Applies to:** Ubuntu Desktop/Server 18.10 to 24.04 and later; KDE neon 2020.03 or later

1. Install Java:

```
sudo apt install openjdk-17-jre
```


Page 30 of 175

2. *(Optional)* Install Git:

```
sudo apt install git
```

3. Install Wine64 by following the Ubuntu Wine guide.
4. Run the Syhunt setup.

Debian

> **Applies to:** Debian 9, 10 or 12 (GUI or minimal)

1. Install Wine64 by following the Debian Wine guide.
2. Install Java (on Debian 9 it is probably already installed):

```
sudo apt install default-jre
```

3. *(Optional)* Install Git:

```
sudo apt install git
```

4. Run the Syhunt setup.

Fedora

> **Applies to:** Fedora 32, 39, 40 and 41

1. Apply all pending updates and reboot.
2. Install Java:

```
sudo yum install java
```

3. *(Optional)* Install Git:

```
sudo yum install git
```

4. Install Wine64 by following the Fedora Wine guide.
5. Run the Syhunt setup.

> **Tip**
> If OpenJDK 17 is installed and the setup fails with an error such as `Exception in thread "main" java.lang.UnsatisfiedLinkError: Can't load library: /usr/lib/jvm/java-17-openjdk-…/lib/libawt_xawt.so`, install the missing OpenJDK packages and run the setup again:
>
> ```
> sudo dnf install java-17-openjdk java-17-openjdk-devel java-17-openjdk-headless
> ```

Syhunt Hybrid 7.2

CentOS and Red Hat Enterprise Linux

> **Applies to:** CentOS 7 or 8 (Everything or Minimal); Red Hat Enterprise Linux

1. Install Wine64 by following the CentOS Wine guide.
2. Install Java:

```
sudo yum -y update
sudo yum install java    # or: sudo yum install java-1.8.0-openjdk
```

3. *(Optional; needed on Minimal for Git scans)* Install Git:

```
sudo yum install git
```

4. *(Minimal installs)* Install wget:

```
sudo yum install wget
```

5. Run the Syhunt setup.

AlmaLinux, Rocky Linux and Oracle Linux

> **Applies to:** AlmaLinux 9, Rocky Linux 9, Oracle Linux 9

1. Install Wine64 by following the Wine guide for your distribution: AlmaLinux, Rocky Linux or Oracle Linux.
2. Install Java:

```
sudo yum -y update
sudo yum install java    # or: sudo yum install java-1.8.0-openjdk
```

3. *(Optional)* Install Git:

```
sudo yum install git
```

4. *(AlmaLinux and Rocky Linux)* Install wget:

```
sudo yum install wget
```

5. Allow unconfined processes to run files that need the SELinux `execmod` permission. This resolves a conflict with Wine; it weakens SELinux slightly but does not disable it.

```
sudo setsebool -P selinuxuser_execmod on
```

6. Run the Syhunt setup.

Amazon Linux 2

1. Install Wine64 by following the Amazon Linux 2 Wine guide.
2. Install Java:

```
sudo yum -y update
sudo yum install java
```


Page 32 of 175

Syhunt Hybrid 7.2

3. *(Optional)* Install Git:

```
sudo yum install git
```

4. Run the Syhunt setup.

Amazon Linux 2023

1. Install Wine64 by following the Amazon Linux 2023 Wine guide.
2. Install Java:

```
sudo yum -y update
sudo yum install java
```

3. *(Optional)* Install Git:

```
sudo yum install git
```

4. Allow the SELinux `execmod` permission, as explained in the AlmaLinux section:

```
sudo setsebool -P selinuxuser_execmod on
```

5. Run the Syhunt setup.

Arch Linux and Manjaro

> **Applies to:** Arch Linux 2019 or later, Manjaro 19

1. Refresh the pacman database:

```
sudo pacman -Syy
```

2. Install Java 8:

```
sudo pacman -S jre8-openjdk
```

3. *(Optional)* Install Git:

```
sudo pacman -S git
```

4. Install Wine64:
   - **Arch Linux:** follow the Arch Linux Wine guide.
   - **Manjaro:** run `sudo pacman -S wine`, then `wine64 ipconfig`. Wine starts and configures itself; press **I** if prompted to install missing components.
5. Run the Syhunt setup.

Linux Mint

> **Applies to:** Linux Mint 18, 19 and 20

1. Java is preinstalled on Linux Mint.

Page 33 of 175

Syhunt Hybrid 7.2

2. *(Optional)* Install Git:

```
sudo apt install git
```

3. Install Wine64 by following the Linux Mint Wine guide.
4. Run the Syhunt setup.

MX Linux

> **Applies to:** MX Linux 19.1 or later

1. Java and Git are preinstalled on MX Linux 19.
2. Install Wine64, then run it once so it configures itself:

```
sudo apt install wine
wine64 ipconfig
```

3. Run the Syhunt setup.

openSUSE

> **Applies to:** openSUSE Leap 15, Leap 42 or later, Tumbleweed

1. *(Optional)* Install Git:

```
zypper in git        # or, if not found: zypper in git-core
```

2. Install Wine64 by following the openSUSE Wine guide.
3. Run the Syhunt setup.

elementary OS

> **Applies to:** elementary OS 5.0 and 5.1

- **elementary OS 5.1:** follow the Ubuntu steps. You may be asked to reboot after installing Wine64.
- **elementary OS 5.0:** read this warning first. If you still want to proceed, you do so at your own risk.

Deepin

> **Applies to:** Deepin 15.9

1. Install Java:

```
sudo apt install default-jre
```

2. *(Optional)* Install Git:

```
sudo apt install git
```


Page 34 of 175

3. Install Wine64, then run it once so it configures itself:

```
sudo apt install wine
wine64 ipconfig
```

4. Run the Syhunt setup.

Solus

> **Applies to:** Solus 4.1

1. Install Java (the `java` command will be at `/usr/lib64/openjdk-8/bin/java` or `/usr/lib64/openjdk-11/bin/java`):

```
sudo eopkg install openjdk-8    # or: sudo eopkg install openjdk-11
```

2. *(Optional)* Install Git:

```
sudo eopkg install git
```

3. Install Wine64 by following the Solus Wine guide.
4. Run the Syhunt setup.

Running the setup (Linux and macOS)

1. Download the Syhunt Hybrid setup with a browser, `wget` or `curl`.
2. Run the setup:

Edition	Command
Hybrid	`java -jar syhunt-hybrid-<version>.jar`
Hybrid on a headless server	`java -jar syhunt-hybrid-<version>.jar -console`
Community	`java -jar <filename>`

   On Linux, if `binfmt-support` is installed (`sudo apt install binfmt-support`), you can instead make the file executable (`chmod a+rx <file>.jar`) and run it directly, for example `./syhunt-hybrid-<version>.jar` (add `-console` on a headless server).

3. Read the EULA carefully before accepting it, then complete the installer (*Next* → *Next* → *Finish*). Syhunt is installed by default in:
   - Linux: `/home/<user>/syhunt-hybrid` or `/home/<user>/syhunt-community`
   - macOS: `/Applications/Syhunt Hybrid/` or `/Applications/Syhunt Community/`
4. Enable the Syhunt Web UI.

Console mode

If the setup runs in console mode:

1. Read the EULA and enter **1** to accept it (or **2** to reject it; Syhunt cannot be installed without accepting it).
2. Press **Enter** to use the default installation path.
3. Enter **1** to continue.
4. Enter **Y** to install Syhunt Core.
5. Enter **1** to continue. The message **Console installation done** confirms the installation.

Syhunt Hybrid 7.2

What's next? Read the Quick Start and the integration guides.

Page 36 of 175

Syhunt Hybrid 7.2

Installing on macOS

> **Applies to:** Syhunt Hybrid 7.2 on macOS Sequoia and earlier

Contents

- Introduction
- Installing on macOS (Apple Silicon or Intel)
- Parallels Desktop: Debian ARM64
- Parallels Desktop: Ubuntu or Kali Linux ARM64
- Parallels Desktop: enabling Rosetta
- Parallels Desktop: installing Wine 8.0

Introduction

Syhunt Hybrid 7.2 supports macOS Sequoia and earlier. The macOS edition is the *Carbon* edition: it is installed on top of Wine and runs on both Apple Silicon Macs (through Rosetta) and Intel Macs.

If you prefer virtualization, Syhunt Hybrid 7.1.6 also runs on ARM64 Linux (Ubuntu, Debian or Kali Linux) inside Parallels Desktop on macOS. This requires a Parallels **Pro** license; the Standard license is not compatible.

Installing on macOS (Apple Silicon or Intel)

1. Read and accept the Syhunt EULA before running any of the commands below.
2. Download Java for macOS (`jre-<version>-macosx-x64.dmg`) and install it, if you haven't already.
3. Install Wine64, if you haven't already:
    - Apple Silicon (M1, M2, M3 or M4)
    - Intel
4. Install wget:

    ```
    brew install wget
    ```

5. Download the Syhunt setup (`.jar` file), Control-click its icon and choose **Open**. Read the EULA carefully before accepting it, then complete the installer (*Next* → *Next* → *Finish*). The Syhunt command-line tools are then ready to use. For more options, see Running the setup.
6. Enable the Syhunt Web UI.

> **Note**
> If, after a macOS upgrade, scanning a Git repository fails with **invalid active developer path**, run `xcode-select --install` to repair the `git` command.

Parallels Desktop: Debian ARM64

> **Applies to:** Debian 12.6 ARM64 on Apple Silicon Macs with Parallels Desktop

1. Install Java:

    ```
    sudo apt install default-jre
    ```


Page 37 of 175

Syhunt Hybrid 7.2

2. *(Optional)* Install Git, if you plan to scan Git repositories:

```
sudo apt install git
```

3. Enable and install Rosetta (see Enabling Rosetta).
4. Install the AMD64 build of Wine64. Debian 12.6 ARM64 installs version 8.0 by default:

```
sudo apt install wine64:amd64
```

5. Run `wine ipconfig` once so Wine configures itself.
6. Install Syhunt: download the `.jar` setup and follow Running the setup.
7. Enable the Syhunt Web UI.

Parallels Desktop: Ubuntu or Kali Linux ARM64

> **Applies to:** Ubuntu 24.04 ARM64 or Kali Linux 2024.2 ARM64 on Apple Silicon Macs with Parallels Desktop

1. Read and accept the Syhunt EULA.
2. Create a dedicated virtual machine for Syhunt.
3. Update the package lists:

```
sudo apt-get update
```

4. Install Java with SVE disabled:

```
# Add this once to the shell profile
echo 'export JAVA_TOOL_OPTIONS="$JAVA_TOOL_OPTIONS -XX:UseSVE=0"' | sudo tee /etc/profile.d/java-
nosve.sh
# Reload the login environment (or open a new terminal)
source /etc/profile.d/java-nosve.sh
# Install OpenJDK 17 and confirm the version
sudo apt install openjdk-17-jre
java --version
```

5. Enable and install Rosetta (see Enabling Rosetta). Skip this step if you're using Parallels' free **Ubuntu with Rosetta** image.
6. The default Wine package is version 9, which doesn't work well with Rosetta. Enable Wine 8.0 first (see Installing Wine 8.0), then install it:

```
sudo apt install libwine:amd64=8.0~repack-4 wine64:amd64=8.0~repack-4
```

   If the `wine` command isn't available afterwards, run `sudo apt install wine`. Confirm the version with `wine --version`.
7. Run `wine ipconfig` once so Wine configures itself.
8. Install Syhunt: download the `.jar` setup and follow Running the setup.
9. Enable the Syhunt Web UI.

Parallels Desktop: enabling Rosetta

> **Applies to:** Ubuntu 24.04 ARM64 or Kali Linux 2024.2 ARM64 on Apple Silicon Macs with Parallels Desktop

1. When the virtual machine has finished installing, shut it down. In the VM settings, go to **Hardware → CPU & Memory**, enable **Use Rosetta to run x86_64 binaries**, then start the VM again.
2. Install Rosetta support:

Page 38 of 175

Syhunt Hybrid 7.2

```
wget https://kb.parallels.com/Attachments/kcs-193687/rosetta_x86_sources.sh
chmod +x rosetta_x86_sources.sh
sudo ./rosetta_x86_sources.sh enable    # prints "AMD64 architecture enabled"
sudo apt update
```

Parallels Desktop: installing Wine 8.0

> **Applies to:** Ubuntu 24.04 ARM64 or Kali Linux 2024.2 ARM64 on Apple Silicon Macs with Parallels Desktop

> **Warning**
> Use this approach in test environments only. It installs Debian 12 (*Bookworm*) packages on Ubuntu 24.04 or Kali Linux 2024.2. It currently works, but mixing packages from different distributions is not a stable long-term solution. We are working on a cleaner method and will update this guide when it's available.

1. Add the Debian 12 repository, pinned so it's used only for Wine:

```
echo 'deb http://deb.debian.org/debian bookworm main contrib non-free non-free-firmware' |
  sudo tee /etc/apt/sources.list.d/debian-bookworm.list

sudo tee /etc/apt/preferences.d/99-wine8 <<'EOF'
Package: wine* libwine* fonts-wine*
Pin: release n=bookworm
Pin-Priority: 990

Package: *
Pin: release n=bookworm
Pin-Priority: 100
EOF
```

2. Add the Bookworm repository signing key:

```
sudo mkdir -p /etc/apt/keyrings &&
curl -fsSL https://ftp-master.debian.org/keys/archive-key-12.asc | gpg --dearmor | sudo tee
/etc/apt/keyrings/debian-bookworm.gpg > /dev/null &&
echo "deb [signed-by=/etc/apt/keyrings/debian-bookworm.gpg] http://deb.debian.org/debian bookworm main"
| sudo tee /etc/apt/sources.list.d/debian-bookworm.list &&
sudo apt update
```

3. Add the Ubuntu repositories for AMD64 packages:

```
sudo tee /etc/apt/sources.list.d/ubuntu-amd64.list > /dev/null <<EOF
deb [arch=amd64] http://archive.ubuntu.com/ubuntu noble main restricted universe multiverse
deb [arch=amd64] http://archive.ubuntu.com/ubuntu noble-updates main restricted universe multiverse
deb [arch=amd64] http://archive.ubuntu.com/ubuntu noble-security main restricted universe multiverse
deb [arch=amd64] http://archive.ubuntu.com/ubuntu noble-backports main restricted universe multiverse
EOF
```

4. Update APT and confirm that Wine 8.0 is available. The output should include `8.0~repack-4`:

```
sudo apt update
apt-cache madison wine | grep 8.0
```

5. Continue with step 6 of the Ubuntu/Kali instructions.

Page 39 of 175

Syhunt Hybrid 7.2

Web UI

> **Applies to:** Syhunt Hybrid 7.2

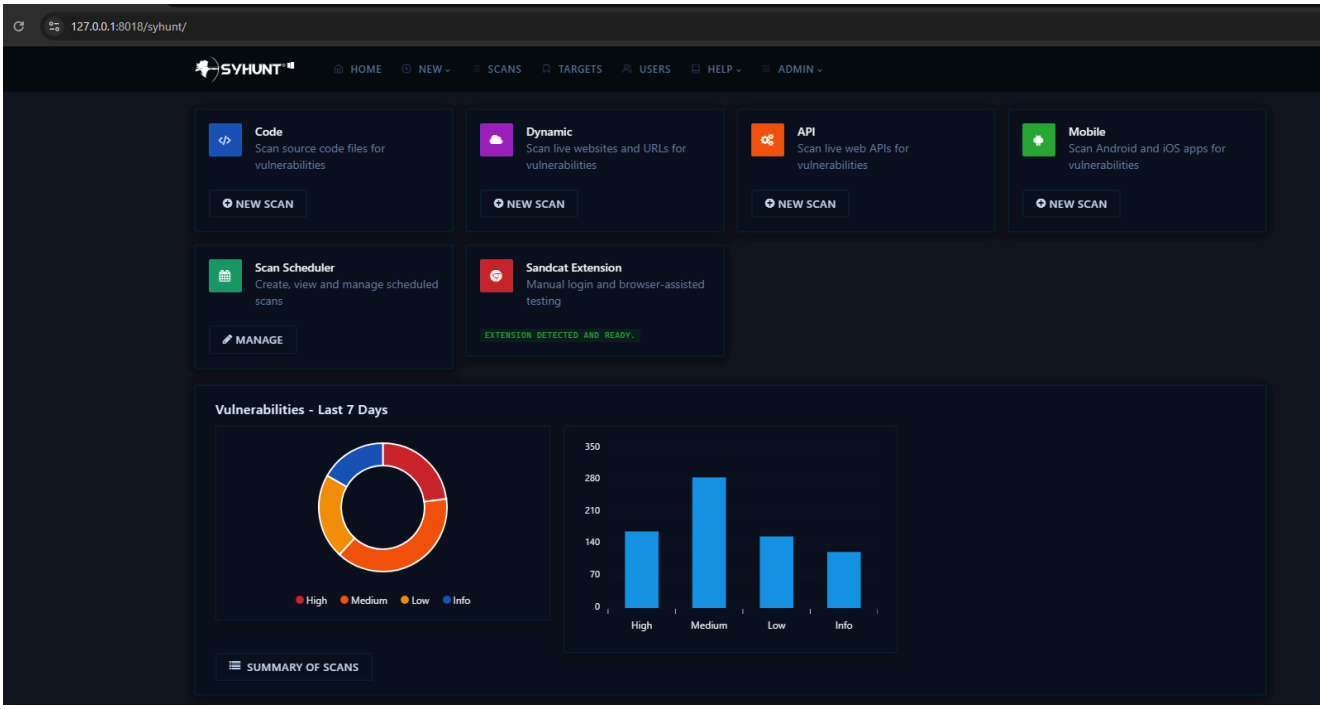
Contents

- Introduction
- Enabling the Web UI
- Creating additional user accounts
- Changing the default UI language
- Password policy
- User roles
- Tips

Introduction

Since version 7.1.6, Syhunt Hybrid includes a web user interface that can be enabled on Windows, Linux and macOS. From it you can launch DAST, SAST and MAST scans; view results of all scans, whether started from the UI, the CLI, the REST API or the scheduler; generate reports and exports; manage users, permissions and active sessions; run console commands; and more. The Web UI and reports are available in eight languages: English, German, Spanish, French, Italian, Japanese, Korean and Portuguese.

The Web UI runs on OpenResty (NGINX + Lua). It hashes passwords with bcrypt at a high cost factor, generates session IDs securely and protects the login against brute-force attacks.



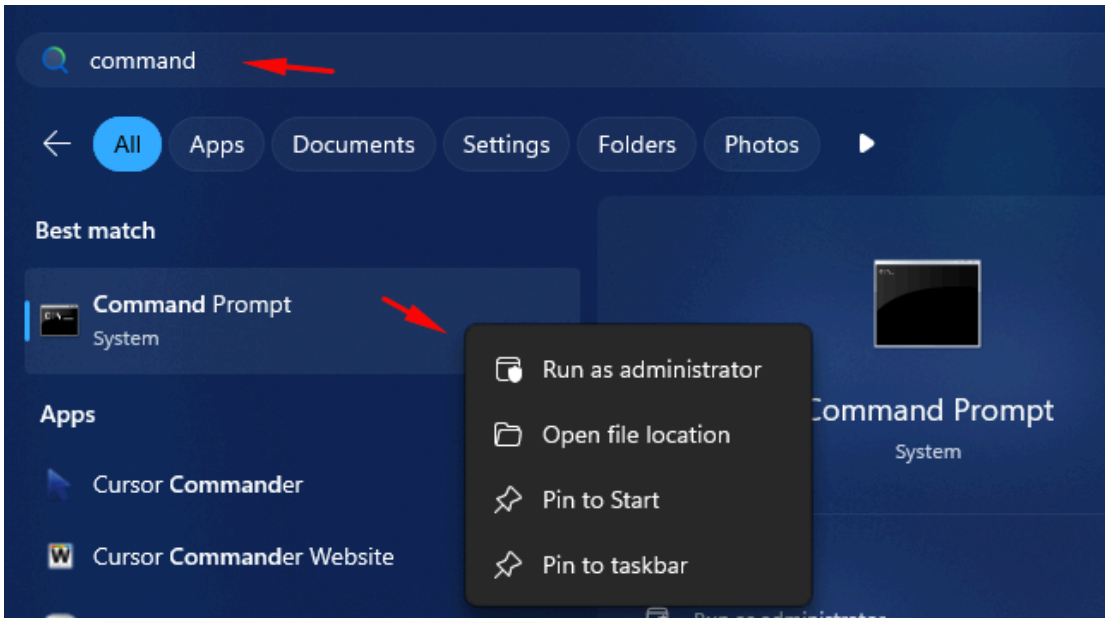
Enabling the Web UI

1. Open a terminal or Command Prompt in the Syhunt CLI folder (the default location depends on your OS):
   - **Linux:** `cd ${HOME}/syhunt-hybrid/carbon/`
   - **macOS:** `cd "/Applications/Syhunt Hybrid/carbon"`

Page 40 of 175

Syhunt Hybrid 7.2

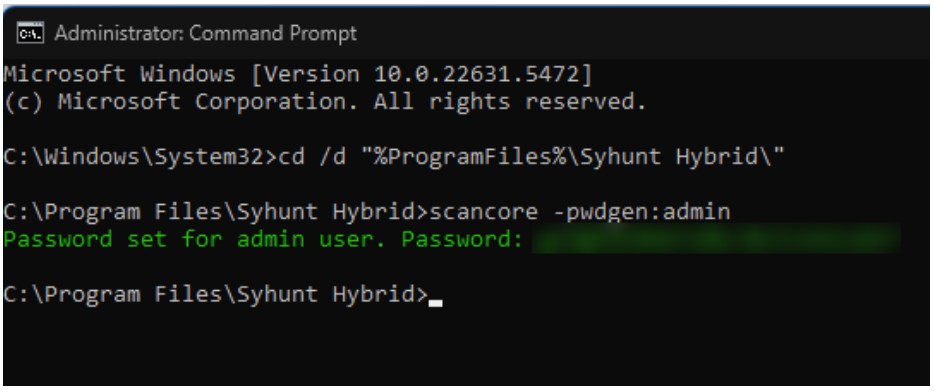
- **Windows:** open the Start menu and type `command`. Right-click **Command Prompt** and choose **Run as administrator**,



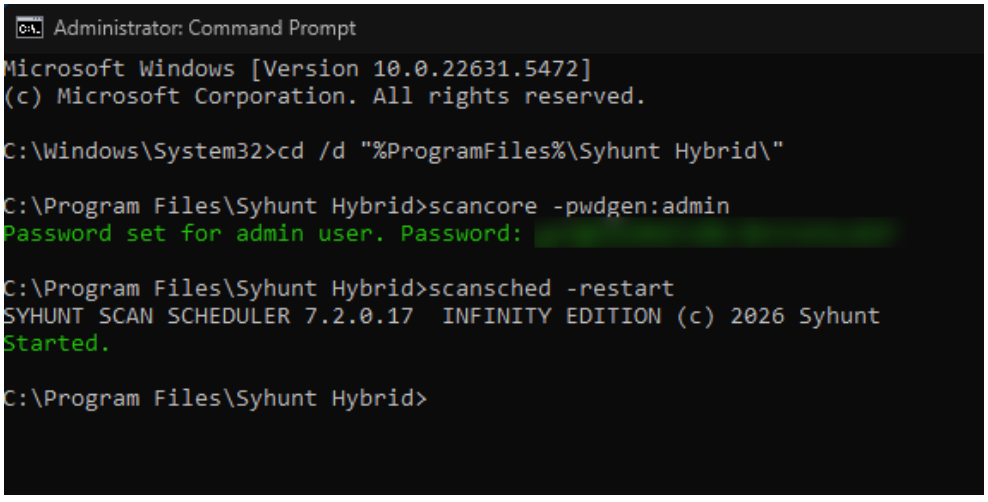
then run `cd /d "%ProgramFiles%\Syhunt Hybrid\"`.

2. Set up the admin account. Run the following command, which generates and sets the admin password, and save the password it prints; you'll need it to log in:

```
scancore -pwdgen:admin
```



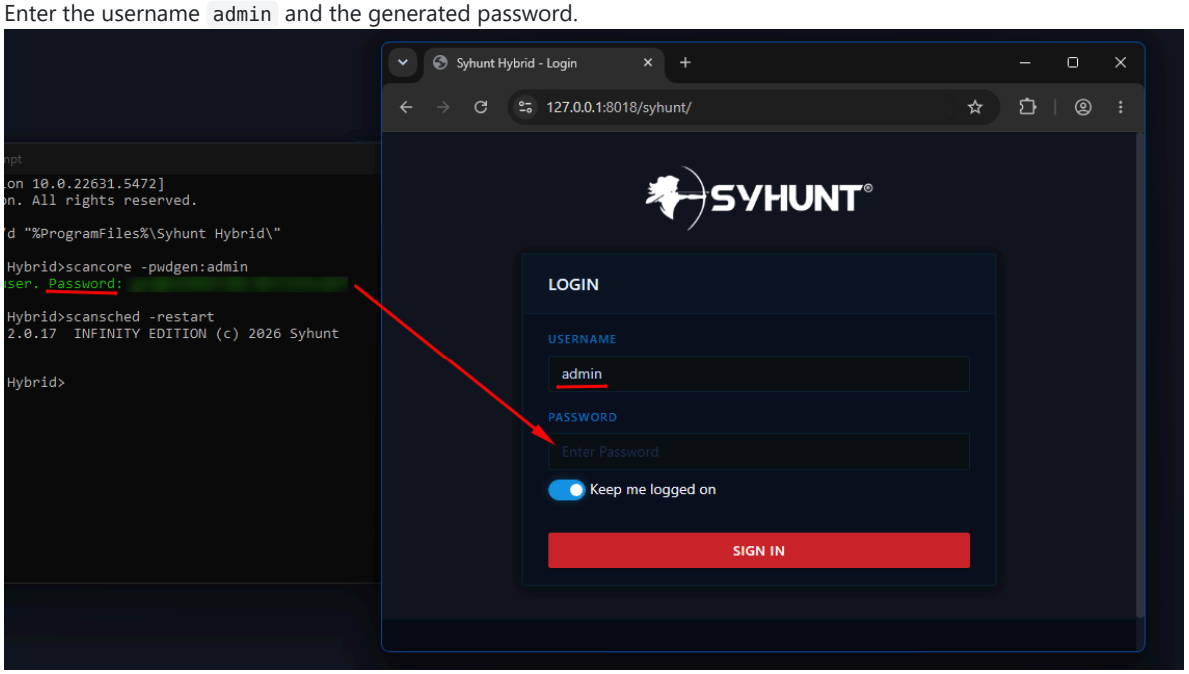
3. Start the server:
   - **Linux or macOS:** run `syservicereg`, then `webui_utils --sslenable`. The Web UI service is installed and started.
   - **Windows:** restart Windows (the Web UI starts at boot), or, to avoid rebooting, run `scansched -restart` as an administrator.



Page 41 of 175

Syhunt Hybrid 7.2

4. Open the Web UI:
    1. In your browser, go to `https://127.0.0.1:8018/syhunt/`.
    2. Enter the username `admin` and the generated password.



3. Click **Sign In**.

Notes

- On **Windows**, once the admin password is set, the Web UI starts whenever the Hybrid Scheduler service starts, so it keeps running even when the current user logs out. On Linux, the Syhunt Web UI service does the same.
- On **AlmaLinux**, if the Web UI service fails to start, you may need to disable SELinux permanently. After rebooting, the service should run normally:

```
sudo sed -i 's/^SELINUX=.*/SELINUX=disabled/' /etc/selinux/config && sudo reboot
```

Creating additional user accounts

To create another account, run:

```
scancore -pwdgen:someusername
```

Changing the default UI language

To change the Web UI's language, run the following command with one of the available language codes:

```
scancore -langset:CODE
```

Password policy

To strengthen security and support compliance, Syhunt 7.1 introduced a strict Web UI password policy:

Requirement	Admin accounts	Other accounts
Minimum length	26 characters	16 characters
Password reuse	Can't reuse the last 24 passwords	Can't reuse the last 10 passwords

Page 42 of 175

Syhunt Hybrid 7.2

Every password must also:

- **not** include the username or any part of the user's full name;
- contain at least one uppercase letter (A–Z), one lowercase letter (a–z), one digit (0–9) and one non-alphanumeric character (such as `@`, `#`, `$`, `%`);
- **not** contain three or more identical consecutive characters (such as `AAA`), or three consecutive ascending or descending characters (such as `123`, `321`, `ABC`, `CBA`).

These rules are checked whenever a password is set. After an upgrade, if an existing password breaks any of them, you'll be asked to reset it at your next login.

The easiest way to create a compliant password is `scancore -pwdgen:<username>`, which securely generates a password, sets it and prints it. To type a password yourself, use `-pwdset` instead.

User roles

When logged in as the administrator, you can assign any user the Operator, Restricted Operator or Viewer role:

Role	Can do
Administrator	Launch scans and view all results, edit user permissions and preferences, and run console commands. Only one admin account is currently allowed.
Operator	Launch scans and view the results of all scans.
Restricted Operator	Launch scans, but view only the results of their own scans.
Viewer	View the results of all scans, but not launch scans.

Permissions by role

Permission	Admin	Operator	Restricted Operator	Viewer
Administrative alerts on the Home screen	✅			
Delete scans, targets, users and connectors	✅			
User access control				
Access the Users screen	✅			
Create users	✅			
Change a user's role	✅			
Access user profiles	✅	Own only	Own only	Own only
Invalidate user sessions	✅	Own only	Own only	Own only
Change a user's password	✅	Own only	Own only	Own only
Scans and targets				
Launch scans	✅	✅	✅	
Access the Targets screen and preferences	✅	✅	✅	✅ (read-only)
Access scan information (logs, comparison, findings, summary, map, reports, exports, debug, list)	✅	✅	Own scans only	✅
Access action information (including preview)	✅	✅	Own scans only	✅
Administrative screens				

Page 43 of 175

Syhunt Hybrid 7.2

Permission	Admin	Operator	Restricted Operator	Viewer
Access the scheduled scan list	✅	✅	✅	✅
Access individual scheduled scan details	✅			
Access the Connectors screen	✅			
Access the command console	✅			
Access the global preferences and lists (including export)	✅			
Access the scheduler log (if enabled)	✅	✅		
Import				
Import API specifications	✅	✅		
Insert or update the product license	✅			

Tips

Scheduler time mismatch

If the time on the Scan Scheduler screen doesn't match your system's time, set the system time zone (replace `Europe/Lisbon` with yours):

- **macOS:** `sudo /usr/sbin/systemsetup -settimezone "Europe/Lisbon"`
- **Linux:** `sudo timedatectl set-timezone Europe/Lisbon`

Scanning private repositories

Before scanning a private Git repository from the Web UI, configure the repository credentials or SSH keys and test the scan with the `scancode` command.

Limitations

Direct connections to non-Git Azure DevOps (TFS) repositories are not supported on Linux or macOS.

Page 44 of 175

Syhunt Hybrid 7.2

Installing Wine

Syhunt Hybrid for Linux and macOS (the *Carbon* edition) runs on Wine64. This page explains how to install Wine64 on each supported system. Once Wine is installed, return to the Linux or macOS guide to install Syhunt.

> **Important**
> Read and accept the Syhunt EULA before running any of the commands below.

Systems: macOS (Apple Silicon) · macOS (Intel) · Ubuntu · Debian / Kali · Fedora · CentOS · AlmaLinux · Rocky Linux · Oracle Linux · Amazon Linux 2 · Amazon Linux 2023 · Arch Linux · Linux Mint · openSUSE · elementary OS · Solus

> **Tip**
> On every system, the last step is to run `wine64 ipconfig` once. Wine starts for the first time and configures itself; if it asks to install missing components (such as Wine Mono or Wine Gecko), press **I** to install them. When it finishes, you'll see a message like `wine: configuration in '/home/<user>/.wine' has been updated`.

macOS (Apple Silicon)

> **Applies to:** macOS Sequoia on Apple Silicon (M1, M2, M3, M4)

1. Install Homebrew, if you haven't already. Open **Terminal** (in `/Applications/Utilities/`) and run:

   ```
   /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
   ```

   Enter your admin password when asked. At the end, Homebrew prints instructions for setting up `~/.zprofile`; follow them.
2. Install XQuartz (you may need to enter your admin password again):

   ```
   brew install --cask xquartz
   ```

3. Install Rosetta:

   ```
   softwareupdate --install-rosetta --agree-to-license
   ```

4. Install Wine:

   ```
   brew install --cask --no-quarantine wine-stable
   ```

   If your Homebrew doesn't support `--no-quarantine`, run these instead:

   ```
   brew install --cask wine-stable
   xattr -dr com.apple.quarantine /Applications/Wine\ Stable.app
   ```

5. After the message `wine-stable was successfully installed!`, run `wine64 ipconfig`.

Page 45 of 175

Syhunt Hybrid 7.2

macOS (Intel)

> **Applies to:** macOS Big Sur and Monterey on Intel

1. Install Homebrew, if you haven't already. Open **Terminal** and run:
   - macOS Catalina and later:

     ```
     /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
     ```

   - macOS High Sierra, Sierra, El Capitan and earlier:

     ```
     /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
     ```

   Enter your admin password when asked, and follow Homebrew's instructions for setting up `~/.zprofile`.
2. Install XQuartz:

   ```
   brew install --cask xquartz
   ```

3. Install Wine (see the Apple Silicon section for the alternative if `--no-quarantine` isn't supported):

   ```
   brew install --cask --no-quarantine wine-stable
   ```

4. Run `wine64 ipconfig`.

Ubuntu

> **Applies to:** Ubuntu 18.10 to 24.04 and later

```
sudo apt install wine-stable
wine ipconfig
```

Syhunt is 64-bit and doesn't need 32-bit Wine. If you want it anyway:

```
sudo dpkg --add-architecture i386 && sudo apt-get update
sudo apt-get install wine32
```

Debian and Kali Linux

> **Applies to:** Debian 9, 10 or 12 (GUI or minimal); Kali Linux 2024.3

```
sudo apt-get update
sudo apt install wine
wine64 ipconfig
```


Page 46 of 175

Syhunt Hybrid 7.2

Fedora

> **Applies to:** Fedora 32, 39 and 40

1. Check for updates:

   ```
   sudo dnf check-update
   ```

2. Run `wine64`. When asked whether to install `wine-core`, type **yes**, then **Y** to confirm the changes.

   If downloading the Wine packages fails, apply all updates, reboot and try again:

   ```
   sudo dnf update
   sudo reboot
   sudo dnf install wine
   ```

3. Run `wine64 ipconfig` and press **I** to install Wine Mono and Wine Gecko when prompted. Wine may print debug errors in the background; you can safely ignore them.

CentOS

> **Applies to:** CentOS 7.7, 7.8, 8.0 and 8.2 (Everything or Minimal)

1. Update the system:

   ```
   sudo yum -y update
   ```

   If this fails with **No URLs in mirrorlist** (CentOS 8 has reached end of life), point the repositories at the CentOS vault. DNF will work again, but you won't receive further updates (details):

   ```
   sed -i 's/mirrorlist/#mirrorlist/g' /etc/yum.repos.d/CentOS-Linux-*
   sed -i 's|#baseurl=http://mirror.centos.org|baseurl=http://vault.centos.org|g' /etc/yum.repos.d/CentOS-Linux-*
   sudo yum -y update
   ```

2. Install EPEL and Wine:

   ```
   sudo yum install epel-release
   sudo yum install wine
   ```

   If Wine64 can't be installed because SDL2 is missing, install SDL2 first and then run `sudo yum install wine` again:

   - Enable the repository that provides SDL2:

     ```
     sudo dnf config-manager --set-enabled PowerTools
     ```

   - CentOS 8.2.3 and later:

     ```
     sudo dnf --enablerepo=PowerTools install SDL2
     ```

   - CentOS 8.4 and later:

     ```
     sudo dnf -y update
     sudo dnf groupinstall 'Development Tools' -y
     sudo dnf config-manager --set-enabled powertools
     ```


Page 47 of 175

Syhunt Hybrid 7.2

```
sudo dnf -y install SDL2-devel
```

3. Run `wine64 ipconfig`.

AlmaLinux and Rocky Linux

> **Applies to:** AlmaLinux 9, Rocky Linux 9

```
sudo yum -y update
sudo dnf install yum-utils
sudo yum install epel-release
sudo dnf config-manager --set-enabled crb
sudo dnf install mingw64-zlib
sudo yum install wine
wine64 ipconfig
```

Oracle Linux

> **Applies to:** Oracle Linux 9

```
sudo yum -y update
sudo yum install epel-release
sudo yum install SDL2-devel
sudo yum install wine
```

If Wine64 can't be installed because nothing provides `mingw64-zlib` and `mesa-libOSMesa`, enable the CodeReady Builder repository, install `mingw64-zlib`, then install Wine again:

```
sudo dnf config-manager --set-enabled ol9_codeready_builder
sudo dnf install mingw64-zlib
sudo yum install wine
```

Finally, run `wine64 ipconfig`.

Amazon Linux 2

```
sudo yum update -y
sudo yum groupinstall "Development Tools" -y
sudo amazon-linux-extras install epel -y
```

Install two key Wine dependencies from compatible Oracle Linux packages, then Wine itself:

```
sudo yum install -y https://yum.oracle.com/repo/OracleLinux/OL8/appstream/x86_64/getPackage/spirv-tools-libs-2023.1-2.el8.x86_64.rpm
sudo yum install -y https://yum.oracle.com/repo/OracleLinux/OL7/9/base/x86_64/getPackage/isdn4k-utils-3.2-99.el7.x86_64.rpm
sudo yum install -y wine
wine64 ipconfig
```

Amazon Linux 2023

1. Add the repositories Wine needs, using Syhunt's helper script:

Page 48 of 175

```
sudo yum groupinstall "Development Tools" -y
wget https://syhunt.fra1.cdn.digitaloceanspaces.com/tools/linux/al2023-createrepos.sh
chmod +x al2023-createrepos.sh
./al2023-createrepos.sh
```

2. Install Wine and run it once:

```
sudo yum update -y
sudo yum install wine -y
wine64 ipconfig
```

3. To prevent conflicts, remove the Oracle Linux 9 repositories added by the script, which are no longer needed:

```
sudo rm /etc/yum.repos.d/epel.repo
sudo rm /etc/yum.repos.d/appstream.repo
sudo rm /etc/yum.repos.d/crb.repo
```

Arch Linux

> **Applies to:** Arch Linux 2019

1. Enable multilib support in pacman: edit `/etc/pacman.conf` (for example, `sudo vi /etc/pacman.conf`) and uncomment these lines, then save the file:

```
[multilib]
Include = /etc/pacman.d/mirrorlist
```

2. *(Optional)* Refresh your pacman keys:

```
sudo pacman-key --init
sudo pacman-key --populate archlinux
sudo pacman-key --keyserver keyserver.ubuntu.com --refresh-keys
sudo pacman -S archlinux-keyring
```

3. Refresh the package database and install Wine:

```
sudo pacman -Syy
sudo pacman -S wine
```

4. Run `wine64 ipconfig` and press **I** to install Wine Mono and Wine Gecko when asked.

Linux Mint

> **Applies to:** Linux Mint 18, 19 and 20

Linux Mint 20

```
sudo apt-get update
sudo apt install wine-stable
wine64 ipconfig
```

Linux Mint 18 and 19

1. Enable the 32-bit architecture (required by the packages, even though Syhunt itself is 64-bit) and install Wine:

Syhunt Hybrid 7.2

```
sudo dpkg --add-architecture i386
sudo apt install wine-stable
```

2. If the installation fails because of dependency problems (as it did in our tests), install Wine from the WineHQ repository instead:

```
wget -nc https://dl.winehq.org/wine-builds/winehq.key; sudo apt-key add winehq.key
sudo apt-add-repository 'deb https://dl.winehq.org/wine-builds/ubuntu/ bionic main'
sudo add-apt-repository ppa:cybermax-dexter/sdl2-backport
sudo apt update && sudo apt install --install-recommends winehq-stable
```

   Use `bionic` for Linux Mint 19.x or `xenial` for Linux Mint 18.x. The PPA provides the required `libfaudio0` library.
3. Run `wine64 ipconfig`.

openSUSE

> **Applies to:** openSUSE Leap 15, Leap 42 or later, Tumbleweed

1. Add the Wine repository for your openSUSE version. The URL pattern is `https://download.opensuse.org/repositories/Emulators:/Wine/openSUSE_<name>_<version>/Emulators:Wine.repo`. For example:

```
# Leap 15.1
sudo zypper ar -r
https://download.opensuse.org/repositories/Emulators:/Wine/openSUSE_Leap_15.1/Emulators:Wine.repo
# Tumbleweed
sudo zypper ar -r
https://download.opensuse.org/repositories/Emulators:/Wine/openSUSE_Tumbleweed/Emulators:Wine.repo
```

2. Refresh the zypper cache. When asked whether to trust the new repository key, enter **a**:

```
sudo zypper ref
```

3. Install Wine (enter **y** to start the installation) and run it once:

```
sudo zypper in wine
wine64 ipconfig
```

elementary OS

> **Applies to:** elementary OS 5.0 and 5.1

- **elementary OS 5.1:** follow the Ubuntu instructions.
- **elementary OS 5.0:** not supported. In our tests, installing Wine left the system unable to boot, so we don't provide instructions for it.

Solus

> **Applies to:** Solus 4.1

Page 50 of 175

Syhunt Hybrid 7.2

> **Warning**
> Solus is not supported. Wine64 was unstable on Solus 4.1 in our tests (errors such as `wine client error:6c: write: Bad file descriptor`), with both the `wine-stable` and `wine-devel` packages.

If you still want to try:

```
sudo eopkg upgrade openssl
sudo eopkg install wine-devel
wine64 ipconfig
```


Page 51 of 175

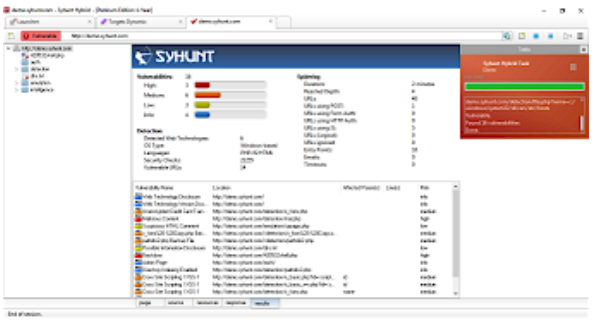
Syhunt Hybrid 7.2

Migration Notes

Read the notes for every version between your current release and the one you are installing.

Migrating to Syhunt 7.2

- **The classic desktop interface has been retired.** The desktop UI shown below is no longer included; it has been replaced by the redesigned Web UI.



- **Feature parity on Linux.** Features previously available only on Windows, such as Selenium-based authentication and the Deep AJAX crawler, are now available on Linux too.
- **Forensic Scanner is CLI-only.** Use the `scanlog` command. A future update will let you view forensic scan results in the Web UI.
- **Review your targets.** Open the Targets screen and look for warning icons next to any target. Hover over an icon to see which preferences or settings need attention. In particular:
    - The **Selenium-Py** authentication method is deprecated. Switch to one of the new methods: Smart, Recorder, Selenium-Auto or Selenium-JS.
    - **AI-Powered Login** has been discontinued. Switch to Smart, Recorder or Selenium-Auto.
- **Plus extensions are built in.** You no longer need to install the separate Plus extensions setup; the 7.2 installer includes every required component.

Migrating to Syhunt 7.0

On Windows Server, make sure the **WebView2 Runtime** is installed. Without it, Syhunt shows a warning when launching its user interface. You can get it from the Evergreen Standalone Installer.

Migrating to Syhunt 6.9.26.1

This release changes how site preferences work in Syhunt Dynamic: different start URL paths on the same domain can now have their own preferences. For example, `https://mydomain/`, `https://mydomain/admin/` and `https://mydomain/members/` are now separate start URLs, each with its own site preferences.

The change applies to preferences set when starting an on-demand scan, in scheduled scans, on the dynamic target screen and through the `scancore` CLI tool.

Review your existing site preferences. If a start URL includes a path (it isn't the site root), its preferences will now show the defaults, and you'll need to re-create any earlier settings such as credentials and crawling restrictions.

Migrating to Syhunt 6.9.9

- On Linux, download and install this update manually instead of using `scanupdate`.

Page 52 of 175

Syhunt Hybrid 7.2

- Check that your scheduled and pipeline code scans of Git repositories target the right branch. Earlier releases defaulted to the `master` branch; from 6.9.9 the branch name is empty by default and the Git client picks the repository's default branch, unless you set one explicitly.
- Make sure you meet the new Internet connection requirements.

Migrating to Syhunt 6.9.7 or later

Up to version 6.9.6, Syhunt for Windows stored user data in `%AppData%\Local\Syhunt`. As part of architecture improvements, version 6.9.7 and later store it in `%ProgramData%\Syhunt`.

Before upgrading, export your preferences so you can import them afterwards:

1. Launch Syhunt Hybrid 6.9.6 (or your older version).
2. Choose ☰ **Menu → Import & Export → Export Preferences**.
3. Save the backup file somewhere easy to find, with a memorable name.
4. Install Syhunt 6.9.7 or later.
5. Choose ☰ **Menu → Import & Export → Import Preferences**, select the backup file and click **Open**.
6. All settings from the earlier installation, including scheduled scans, are now imported and active.

(Optional) Migrating session history and output

- To keep your past sessions, move the files and folders from `%AppData%\Local\Syhunt\Sessions` to `%ProgramData%\Syhunt\Sessions`. They will appear again on the Past Sessions screen.
- If needed, also move `%AppData%\Local\Syhunt\Output` to `%ProgramData%\Syhunt\Output`.
- The other folders under `%AppData%\Local\Syhunt\` contain temporary data and can be safely deleted.

Forced AppData mode

On Windows, Syhunt 6.9.7.1 and later can be forced to keep using the local AppData folder instead of ProgramData: create an empty file named `LocalAppData.json` in the default Syhunt location (see the Command Line guide for the default locations).

> **Warning**
> Some key features, such as the scan scheduler, do **not** work in this mode.

Page 53 of 175

Syhunt Hybrid 7.2

Performance Tuning

Dealing with long DAST scans

Scanning a website that generates a large amount of dynamic content can take a long time with Syhunt Dynamic. This isn't a bug or a limitation, but a consequence of a design choice: Syhunt Dynamic does **not** limit how deep it goes when mapping a site's structure and entry points, because the goal is always to test the target fully. Some sites, however, are extremely large or generate a practically infinite number of links, which becomes a problem when a scan must finish within a set time.

We consider that a DAST scan against a site with heavily dynamic content should take no more than **three hours**. To get there, Syhunt Dynamic offers:

- **Incremental scanning**, which shortens repeat scans of the same target.
- **A large and growing set of built-in rules** that speed up scans of such sites. New rules ship with Syhunt updates every week.

DAST optimization

When the built-in rules aren't enough for a particular website, a Syhunt expert can analyze it on request and add custom rules. We call this process *DAST optimization*; it usually takes just a few hours.

The new rules speed up mapping and guide the scanner toward the relevant areas and entry points of the site, skipping redundant dynamic content. No attacks are launched against the website during optimization, so there's no risk to its integrity or availability. When the work is done, a new Syhunt release containing the new rules is made available.

To request it, open a support ticket listing the target URLs that need DAST optimization.

Complementing DAST with SAST

For websites with heavily dynamic content, Syhunt recommends also scanning the source code with Syhunt Code. Static analysis covers code paths that are hard to reach by crawling.

Speeding up code scans

Real-time antivirus protection can slow down source code scans considerably. See Preventing antivirus interference in pipelines.

Page 54 of 175

Syhunt Hybrid 7.2

Scanning

Page 55 of 175

Syhunt Hybrid 7.2

Dynamic Scanning (DAST)

> **Applies to:** Syhunt Dynamic 7.2

Contents

- How to perform a dynamic scan
- Logging in manually through the browser
- Scanning from the command line
- Scanning IPv6 addresses
- Preventing a vulnerability from being reported
- Pre-scan requirements
- Related guides

How to perform a dynamic scan

In a standard dynamic scan (also known as a black-box scan), Syhunt injects data into the web application and then analyzes the responses to determine whether the code is vulnerable to specific web application attacks.

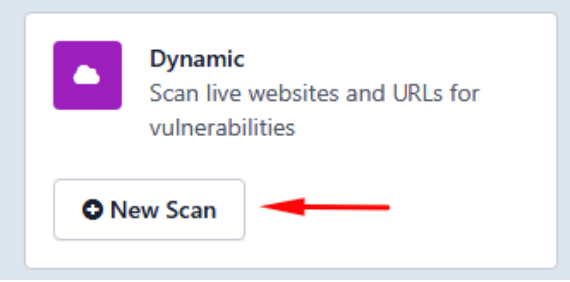
Main supported languages

ASP (Classic) · ASP.NET · Java / JSP · JavaScript · Lua · Perl · PHP · Python · Ruby

Running the scan

Follow these steps to run a dynamic scan and generate a vulnerability report:

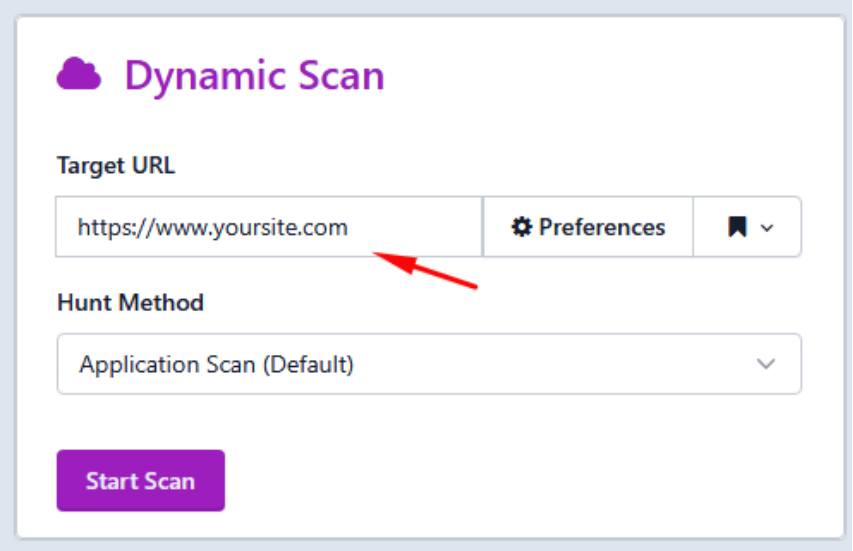
1. ⚠️ Make sure you meet the pre-scan requirements and are authorized to scan the target.
2. Enable the Web UI, if you haven't done so yet.
3. In the Syhunt Web UI, click Dynamic's **New Scan** button.



Page 56 of 175

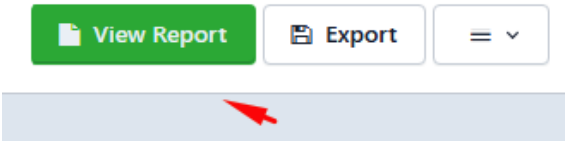
Syhunt Hybrid 7.2

4. Enter the URL of the website you want to scan.



5. Select a scan method. We recommend **Application Scan (Default)**, which checks for all vulnerabilities using the recommended settings. The methods are explained in Hunt Methods.
6. *(Optional)* Click **Preferences**. On the **Technologies** tab, select the technologies used by the target website. You can change other site-specific preferences on the same screen. Review the settings and click **Save**.
7. Click **Start Scan**.

When the scan finishes, click **View Report** to open the results as an HTML report, or **Export** to download them in your preferred format.



Logging in manually through the browser

See How to perform a manual authenticated scan with Syhunt Sandcat.

Scanning from the command line

See Command Line: dynamic scans.

Scanning IPv6 addresses

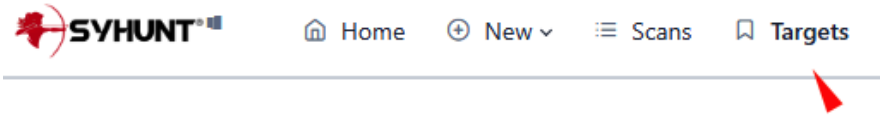
Syhunt Dynamic fully supports IPv6 targets. Enclose the address in square brackets, for example:

```
http://[2001:4860:0:2001::68]/index.php
```

Preventing a vulnerability from being reported

You can stop specific vulnerabilities from being reported by using ignore IDs or ignore rules:

1. Open **Targets** in the menu

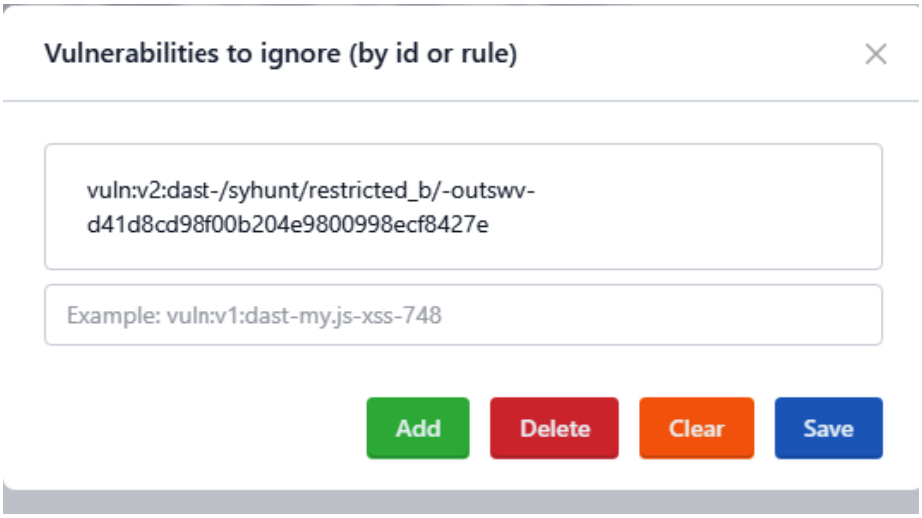


   and click the target in the list.
2. Go to the **Manage** tab.

Page 57 of 175

Syhunt Hybrid 7.2

3. In the **Exclusions** group, click **Vulnerabilities**.
4. Type the ignore ID or rule and click **Add**.



Each vulnerability entry in a report ends with its ignore ID. Ignore IDs are the simplest and recommended way to suppress a finding. For broader scenarios, you can write ignore rules instead.

For common questions, see the FAQ.

Pre-scan requirements

1. ⚠️ **Only system administrators, or other people in charge of the target, should use this software. Do not scan websites outside your direct control.**
   - If you need to scan a website you don't control, obtain written permission from its owner or administrator first.
   - As good practice, back up the website's source code and database before scanning. In rare cases, injection testing against an insecure, fragile website can pollute the database or alter files in ways that affect how the site works.
2. Make sure you meet the Internet connection requirements.
3. Read and accept the Syhunt EULA before launching any scan.

Related guides

- Hunt Methods: choose the right scan method.
- Authenticated Scanning: scan areas behind a login.
- Scheduling Scans: run scans automatically.
- Reporting: customize reports.
- Technical Specifications: system requirements.

Page 58 of 175

Syhunt Hybrid 7.2

Authenticated Scanning

> **Applies to:** Syhunt Dynamic 7.2.0.24

Contents

- Introduction
- Available authentication methods
- What's new in version 7.2
- Configuring and testing authentication
- Recording a login sequence (Recorder)
- Scanning with a manual login (Sandcat)
- Selenium-based authentication
- Scheduled scans with authentication
- Authenticated scans from the command line
- Customizing form auto-completion
- Preventing accidental logout
- Using client SSL certificates

Introduction

Since version 7.2, Syhunt Dynamic supports a wide range of authentication methods: **Standard**, **Recorder**, **Selenium-Auto**, **Selenium-JS**, **Smart**, **Basic**, **Bearer**, **Digest**, **NTLM** and **Manual**.

Most methods are automated: Syhunt Dynamic logs in using the credentials and settings configured on the Target Preferences screen. Depending on the method, it detects the login form automatically, replays a recorded browser login, runs a Selenium script or uses an HTTP authentication mechanism. Where the method supports it, Syhunt Dynamic monitors the authenticated session and logs in again automatically if the session is lost.

Manual authentication starts a scan from a browser session you've already logged into in Edge or Chrome, using the Syhunt Sandcat browser extension. Syhunt Dynamic monitors that session throughout the scan and reports whether it was lost before the scan finished.

Available authentication methods

Method	Description
None	Authentication is disabled.
Standard	Classic form-based login using direct HTTP requests, without a browser session.
Smart	Tries Selenium-Auto first and falls back to Standard if needed.
Recorder	Replays a login sequence recorded in Edge, Chrome or another Chromium-based browser.
Pre (Manual)	Uses a browser session you logged into in Edge or Chrome, through the Syhunt Sandcat extension. Unlike the other methods, it's started from Sandcat rather than configured on the Target Preferences screen.
Selenium-Auto	Automatically generates and runs a Selenium login script. No coding required.
Selenium-JS	Runs a custom Node.js login script that you provide. For security reasons, scripts can only be entered through the CLI.
Bearer	Authenticates requests with a bearer token. Common in REST services, web APIs, mobile apps and modern web applications.
Basic	HTTP Basic authentication.
Digest	HTTP Digest authentication.

Page 59 of 175

Method	Description
NTLM	NTLM authentication, common in Windows and Active Directory environments.

What's new in version 7.2

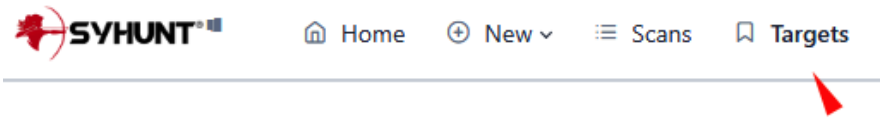
Version 7.2 introduces **Syhunt Authenticator 2.0**, a major redesign of Syhunt Dynamic's authentication framework.

- **One module for every method.** The authenticator used to handle mainly Selenium-based logins. It now manages all authentication methods, for a more consistent configuration experience.
- **One place to configure it.** Settings previously split between web form authentication and server-side methods (Basic, Digest, NTLM) are now together on the Target Preferences screen.
- **New methods:**
  - **Recorder**: records and replays browser-based login sequences.
  - **Smart**: combines Selenium-Auto and Standard for better compatibility.
  - **Selenium-Auto**: generates and runs Selenium login scripts with no code to write.
  - **Selenium-JS**: runs custom login scripts written in Node.js.
- **Removed methods.** The legacy **Selenium-Python** and **AI-Powered** methods are gone. You no longer need to install and maintain a separate Python environment and third-party dependencies, which also reduces exposure to external components and software supply-chain attacks. Selenium-Auto replaces AI-Powered login and works in much the same way, without requiring Python or an AI integration.

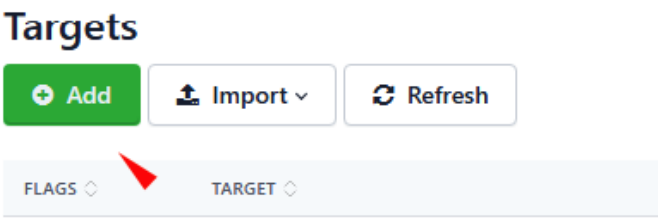
The result is an authentication setup that is simpler, more secure and better integrated with the Syhunt Dynamic scanning workflow.

Configuring and testing authentication

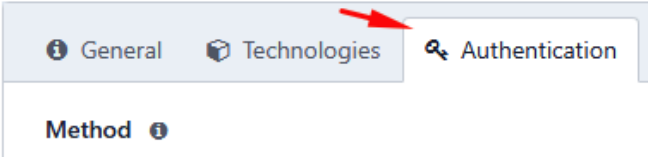
1. Open the **Targets** screen in the Web UI.



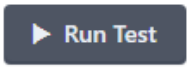
2. Add the URL to the list of targets, if it isn't there yet.



3. Open the target and go to the **Authentication** tab.



4. Select the authentication method.
5. Enter the credentials.
6. Click **Run Test** to check that authentication works.



Syhunt Hybrid 7.2

- On success, you'll see a **Success** message and a screenshot of the page after login.

- On failure, you'll see a **Failed** message and a debug log.

For method-specific instructions, see Recorder, Selenium or Manual login.

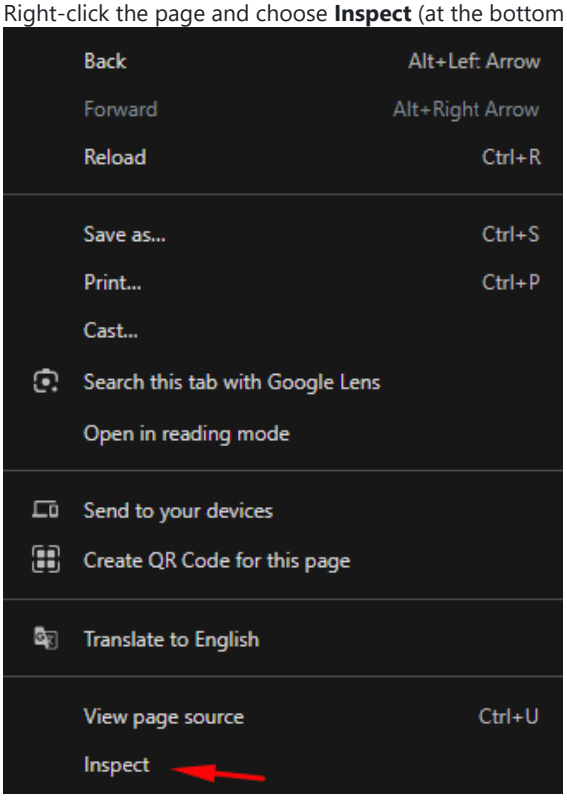
Recording a login sequence (Recorder)

With the **Recorder** method, Syhunt Dynamic logs in by replaying a login you recorded in Edge, Chrome or another Chromium-based browser. Instead of writing a script, you record yourself logging in, export the recording as JSON and paste it into the Target Preferences screen. Syhunt Dynamic replays the recorded steps whenever it needs to log in during a scan.

> **Note**
> **Linux users:** don't use Backspace to correct typing mistakes while recording. If you mistype, start the recording over. In our tests, recordings containing Backspace keystrokes sometimes failed to replay on Linux.

1. Record the login

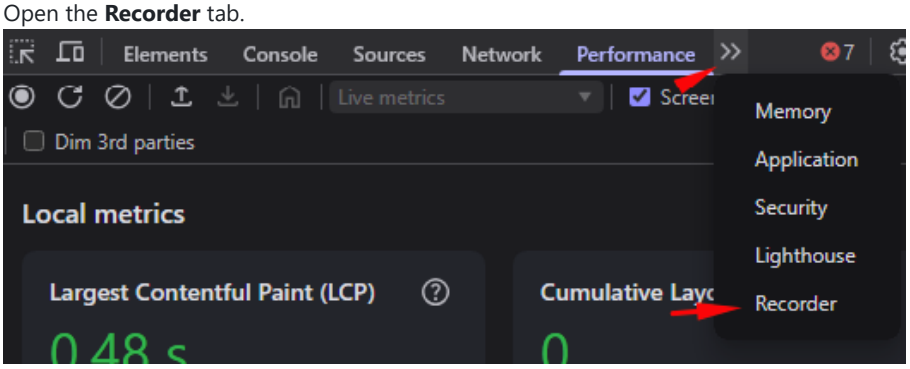
1. In Edge, Chrome or another Chromium-based browser, go to the website you want to scan.
2. Right-click the page and choose **Inspect** (at the bottom of the menu).



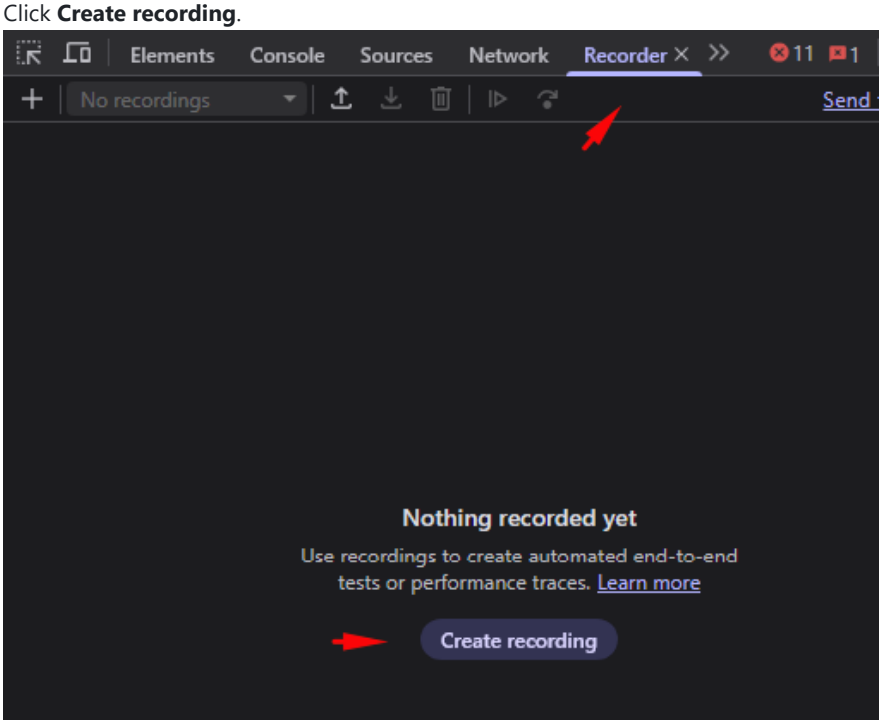
Page 61 of 175

Syhunt Hybrid 7.2

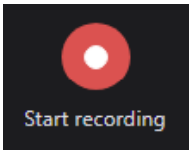
3. Open the **Recorder** tab.



4. Click **Create recording**.

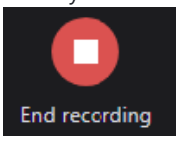


5. Enter a name for the recording and click the red **Start recording** button.

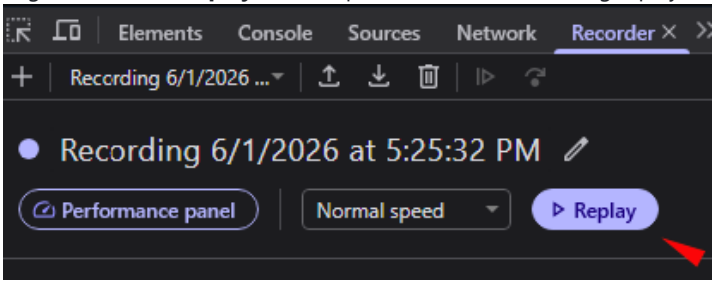


6. Log in as usual.

7. When you're done, click **End recording**.



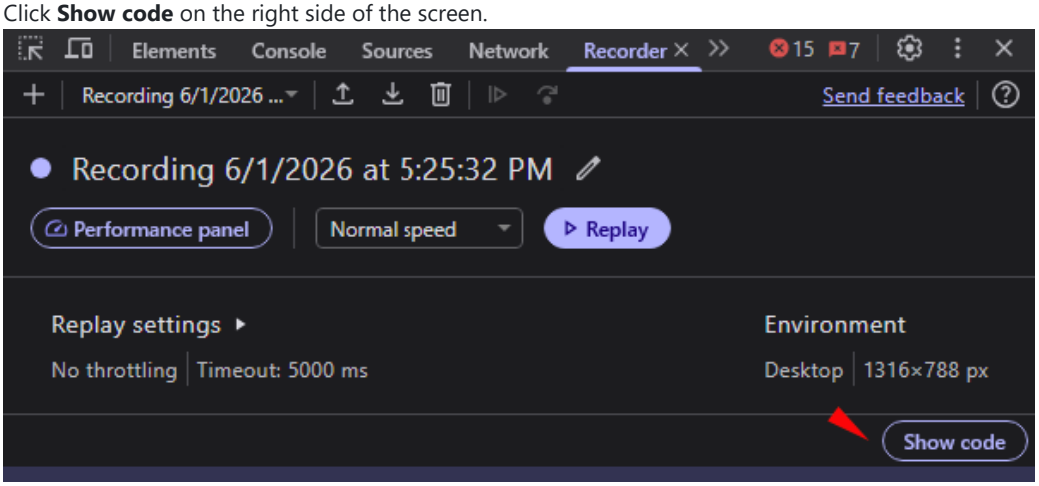
8. Log out and click **Replay** at the top to confirm the recording replays correctly.



Page 62 of 175

Syhunt Hybrid 7.2

9. Click **Show code** on the right side of the screen.

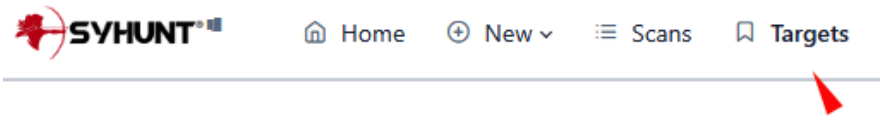


10. Select the JSON code manually, right-click it and choose **Copy**.

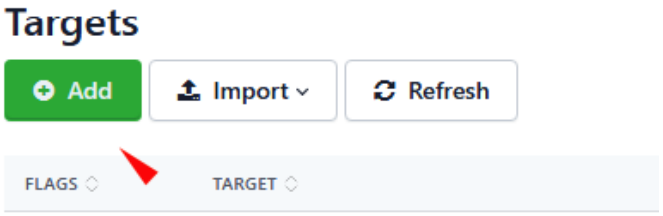
> **Tip**
> Don't use *Select all* followed by *Copy*: a small Chrome UI bug prevents the whole JSON from being selected.

2. Add the recording to Syhunt

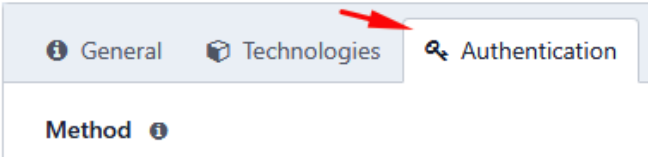
1. Open the **Targets** screen in the Web UI.



2. Add the login URL to the list of targets, if it isn't there yet.



3. Open the target and go to the **Authentication** tab.

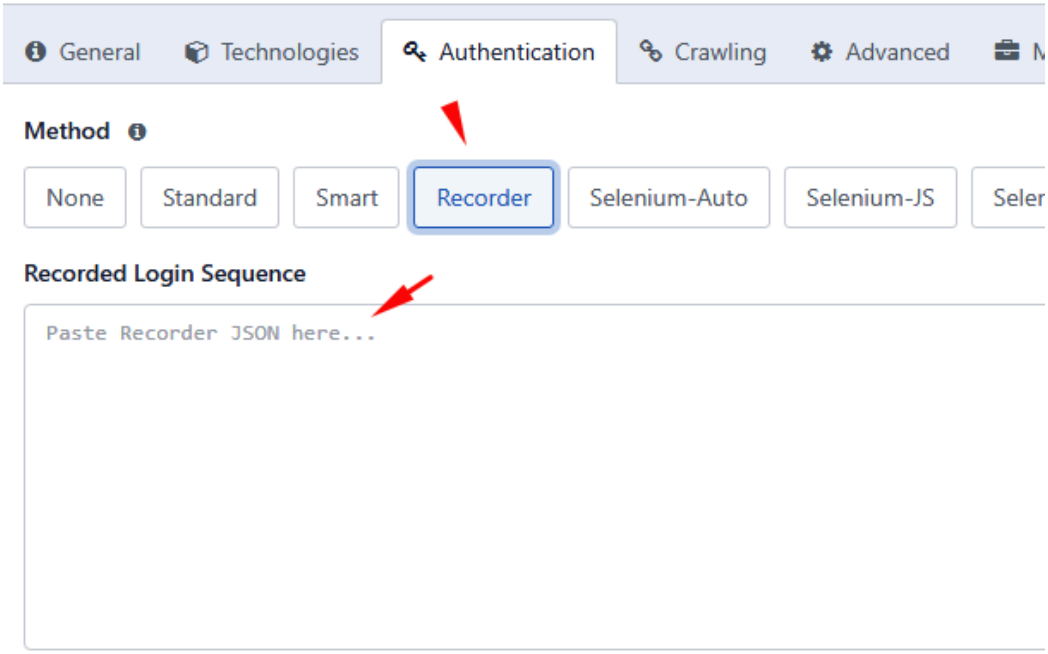


4. Select **Recorder**.

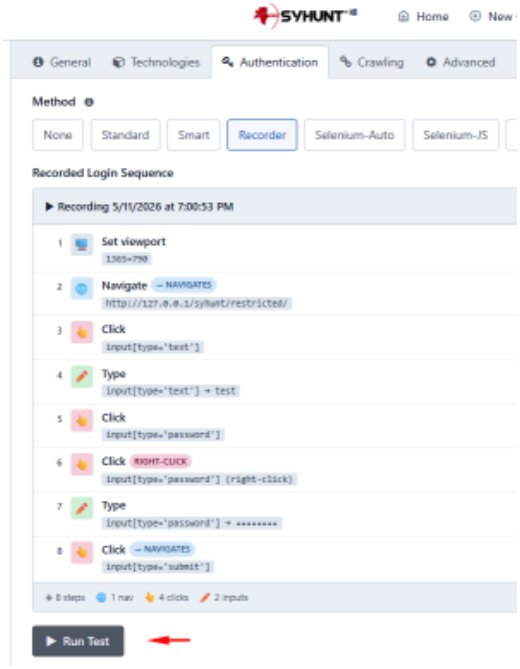
Page 63 of 175

Syhunt Hybrid 7.2

5. Paste the JSON into the **Recorded Login Sequence** field.



6. Click **Run Test** to confirm that Syhunt can replay the recording.



Scanning with a manual login (Sandcat)

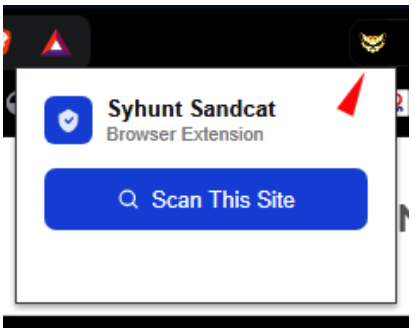
With the Syhunt Sandcat extension installed in your browser (Edge, Chrome, Brave or another Chromium-based browser), you can start a scan directly from the browser, using the session you established by logging in to the target website.

1. Make sure the Syhunt Sandcat extension is installed and enabled.
2. In the browser, go to the website you want to scan.
3. Log in with your credentials.

Page 64 of 175

Syhunt Hybrid 7.2

4. Click the Sandcat icon in the browser toolbar, then **Scan This Site**.



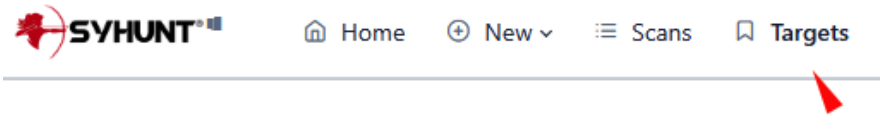
5. A new screen opens. A green **Session Ready** message confirms that the session was captured.
6. Click **Start Scan**.

The session details section of the report shows whether the manually started session lasted from the beginning to the end of the scan (**Authenticated Session Maintained: Yes**).

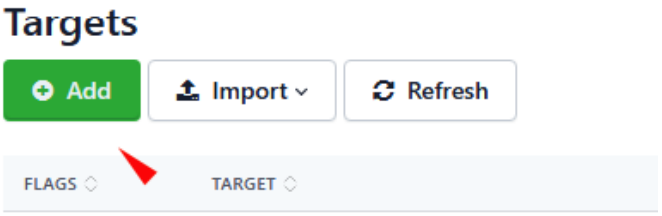
Selenium-based authentication

1. Configure the target

1. Open **Targets** from the top menu.



2. Add the URL to the list of targets, if it isn't there yet.



3. Click the URL. The Target Preferences screen opens.
4. On the **Authentication** tab, select **Selenium-Auto** or **Selenium-JS**. Selenium-Auto generates the script for you; Selenium-JS runs a script you write.
5. Enter the username and password.
6. Click **Save**. The Target Preferences window closes.
7. If you chose **Selenium-JS**, enter your script through the CLI, for example:

```
scancore -tg:http://yourtargetapp.com -prefset:dynamic.formauth.script.seleniumjs.encrypted -
fromfile:path/to/myseleniumscript.js
```

When writing the script:

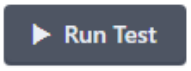
- Start at the part that fills in and submits the login form.
- Don't call `driver.get()` at the start or `driver.quit()` at the end; Syhunt handles both.
- Use the placeholders `@user@` and `@pass@` for the username and password entered on the Authentication tab, for example `username_field.sendKeys('@user@')` and `password_field.sendKeys('@pass@')`.

2. Test it

Page 65 of 175

Syhunt Hybrid 7.2

1. Click the configured URL again and click **Run Test**.



2. If you see **Successful login** and a screenshot of the logged-in page at the top of the screen, the configuration is ready for future scans.
3. If it keeps failing after several retries, adjust your Selenium script and try again.

Reports from authenticated scans include, at the top, a screenshot of the target after login and an indication of whether the session was maintained from start to finish.

Scheduled scans with authentication

1. Open the **Scans** screen and click **Scheduled Scans**.
2. Click a scheduled dynamic scan, then the **Preferences** button next to the target field.
3. On the **Authentication** tab, choose the authentication method and enter the credentials.

Authenticated scans from the command line

There are two approaches:

- **A.** Pass the credentials to `scanurl` when you launch the scan.
- **B.** Store the credentials beforehand with `scancore`.

Method B is more secure, because the password isn't exposed on the `scanurl` command line.

```
-- Method A: form authentication specified when launching the scan.
-- The credentials are not stored in the Dynamic settings.
scanurl http://127.0.0.1 -atype:Form -auser:yourusername -apass:yourpassword

-- Method B: authentication stored for a specific website.
-- The credentials are saved in the Dynamic settings and reused in every scan of this target.
-- Available types: None, Smart, Standard, Basic, Bearer, Digest and Selenium-Auto.
-- None disables authentication.
scancore -tg:http://127.0.0.1 -prefset:dynamic.formauth.type -v:Standard
scancore -tg:http://127.0.0.1 -prefset:dynamic.formauth.username -v:myuser
scancore -tg:http://127.0.0.1 -prefset:dynamic.formauth.password -vsecret
-- For Bearer authentication, set the token instead:
scancore -tg:http://127.0.0.1 -prefset:dynamic.formauth.token -vsecret
scanurl http://127.0.0.1
```

`-vsecret` prompts you (twice, with masked input) for the value, so it never appears on the command line. See Preferences.

Customizing form auto-completion

> **Note**
> This applies to the **Standard**, **Smart** and **Selenium-Auto** methods, and is mainly intended for advanced scenarios.

Syhunt Dynamic detects and fills in login form fields automatically in most applications. You only need to configure fields manually when a form uses non-standard field names, custom controls or extra fields that can't be identified automatically.

For example, this form has an extra hidden field and non-standard names:

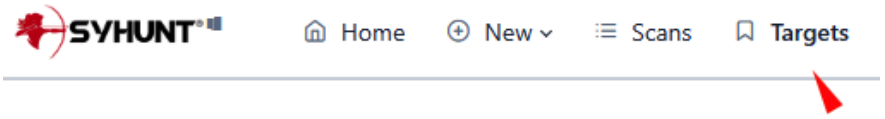
```
<input name="ClientUTBox" id="ClientUTBox" type="hidden" value="1234">
<input name="ClientUNBox" id="ClientUNBox" type="text" class="InputBox"/>User Name
<input name="ClientPWBox" id="ClientPWBox" type="password" class="InputBox">Password
```

To tell Syhunt how to fill it in:

Page 66 of 175

Syhunt Hybrid 7.2

1. Open **Targets** from the top menu.



2. Click the target URL (it must already be in the list). The Target Preferences screen opens.
3. On the **Authentication** tab, click **Optional** to show the additional options.
4. Click **Custom Fields** and add these lines:

```
ClientUTBox=1234
ClientUNBox=@syhunt_web_form_username
ClientPWBox=@syhunt_web_form_password
```

5. Click **Save** to save the custom fields, then **Save** at the bottom of the screen.

Alternatively, set the same fields from the command line:

```
scancore -tg:"[targeturl]" -prefset:dynamic.formauth.fields.encrypted -v:[ClientUTBox=1234],
[ClientUNBox=@syhunt_web_form_username],[ClientPWBox=@syhunt_web_form_password]
```

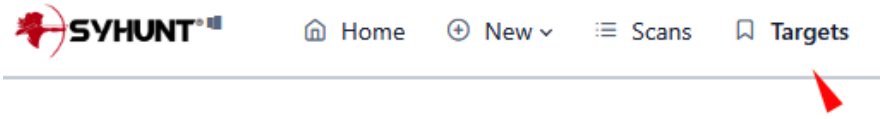
Values that start with `@` are internal variables: they insert the username and password entered on the Target Preferences screen into those fields. When you're done, click **Run Test** to confirm that login works with the configured credentials and custom fields.

Preventing accidental logout

Syhunt Dynamic detects most logout pages and avoids them while crawling. If it does hit one, it can usually log in again automatically and carry on.

For applications with custom logout URLs or unusual logout mechanisms, you can list the logout pages so the crawler avoids them altogether:

1. Open the **Targets** screen.



2. Open the target.
3. Go to the **Crawling** tab.
4. Under **Exclusions**, click **Logout URLs** and add your logout URL, for example `/getmeout.php`.
5. Click **Save**. The dialog closes.

Using client SSL certificates

If the target requires client SSL certificates, configure them as follows:

1. Open **Admin** → **Connectors**.
2. Click **Add**.
3. Set **Type** to **Web Certificate**, enter a name for the certificate and click **Create**.
4. Click the new certificate in the list. The Web Certificate screen opens.
5. Select the certificate files and click **Save**.
6. Open the **Targets** screen and click the website that needs the certificate. The Target Preferences screen opens.
7. On the **Advanced** tab, under **Certificate**, select the certificate from the list.

Page 67 of 175

Syhunt Hybrid 7.2

Source Code Scanning (SAST/MAST)

> **Applies to:** Syhunt Code 7.2

Contents

- How to perform a code scan
- Supported languages
- Scanning from the command line
- Preventing antivirus interference in pipelines
- Preventing a vulnerability from being reported
- Related guides

How to perform a code scan

Syhunt's white-box scan (source code scan) uncovers many classes of application vulnerabilities and highlights areas of the code that need review. Its static analysis detects a very large number of vulnerability types, including the 2025 CWE Top 25 Most Dangerous Software Weaknesses and the OWASP Mobile Top 10 risks, across many web and mobile programming languages.

Supported languages: web

Language	Frameworks and variants
C#	Blazor, Razor Pages, ASP.NET Core MVC, ASP.NET Web Forms
ASP Classic	VBScript and JavaScript
Dart	
Java	JEE / JSP
JavaScript	Client and server side, Node.js, Angular, AngularJS, React, Express.js, Koa.js
Kotlin	Ktor
Lua	ngx_lua, mod_lua, CGILua, Lua Pages
Object Pascal	Delphi XE and older, Lazarus, DWS
Perl	Mojolicious, PSGI/Plack, CGI
PHP	
Python	CGI, Django, mod_python, WSGI
Ruby	Rails, ERB
TypeScript	Client and server side, Node.js, Angular
VB	VB.NET, ASP.NET Web Forms

Supported languages: mobile

Language	Platform
Java	Android
Kotlin	Android
Swift	iOS
Objective-C, C and C++	iOS

Page 68 of 175

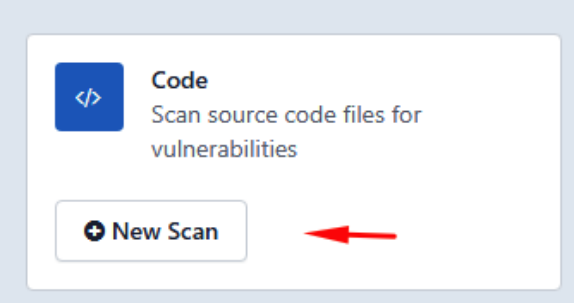
Syhunt Hybrid 7.2

Language	Platform
C#	.NET MAUI, Xamarin (Android/iOS)
Dart	Flutter (Android/iOS)
Object Pascal	Delphi XE
JavaScript	Including Node.js, Angular, AngularJS, Express.js, Koa.js

Running the scan

Follow these steps to run a source code scan and generate a vulnerability report:

1. Enable the Web UI, if you haven't done so yet.
2. In the Syhunt Web UI, click Code's **New Scan** button.



3. Enter the project URL, or the file to scan.
4. *(Optional)* If you entered a project URL, you can also enter a branch name. If you leave it blank, the main branch is used.
5. Select a scan method. We recommend **Application Code Scan (Default)**, which checks for all vulnerabilities using the recommended settings. The methods are explained in Hunt Methods.
6. Click **Start Scan**.

When the scan finishes, click **View Report** to open the results as an HTML report, or **Export** to download them in your preferred format.

Scanning from the command line

See Command Line: code scans.

Preventing antivirus interference in pipelines

When analyzing a large codebase, Syhunt opens every file in the repository. If real-time antivirus protection is enabled, the antivirus also scans each of those files, which can slow the scan down considerably.

In our tests on a machine running Microsoft Defender, scanning a PHP repository of about 10,000 files took 10 minutes longer and used 25% more CPU because of the Antimalware Service (`MsMpEng.exe`). In a CI/CD pipeline, those 10 minutes matter.

The fix is to add a **Process** exclusion in Defender. A process exclusion stops the antivirus from scanning the files that the Syhunt process opens; a *file* exclusion, by contrast, only excludes the executable itself.

To add a process exclusion in Microsoft Defender:

1. Open **Start** → **Settings**.
2. Under **Privacy & security**, select **Virus & threat protection**.
3. Under **Virus & threat protection settings**, select **Manage settings**.

Page 69 of 175

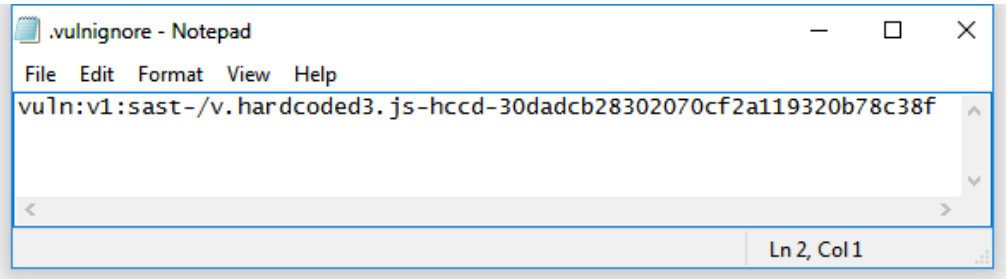
Syhunt Hybrid 7.2

4. Under **Exclusions**, select **Add or remove exclusions**, then choose **Process**.
5. Enter the full path of the Syhunt Code scanner, `C:\Program Files\Syhunt Hybrid\ScanCode.exe`, and click **Add**.

If you use another antivirus product, see its documentation for the equivalent setting.

Preventing a vulnerability from being reported

To suppress specific findings, create a `.vulnignore` file in the root of the repository or directory being scanned. Put one ignore ID or ignore rule per line.



Each vulnerability entry in a report ends with its ignore ID. Ignore IDs are the simplest and recommended way to suppress a finding. For broader scenarios, you can write ignore rules instead.

Related guides

- Hunt Methods: choose the right scan method.
- Scheduling Scans: run scans automatically.
- Reporting: customize reports.
- Technical Specifications: system requirements.

Page 70 of 175

Syhunt Hybrid 7.2

Web API Scanning

> **Applies to:** Syhunt API 7.2

Contents

- How API scanning works
- Supported specification formats
- Running an API scan
- Demo specifications
- Preventing a vulnerability from being reported
- Hunt methods for APIs
- Scheduling API scans
- Pre-scan requirements
- Hosting and uploading specifications
- Safeguards against accidental scanning
- Specification format requirements

How API scanning works

An API scan works much like a traditional dynamic scan: Syhunt injects test data into the API's endpoints and analyzes the responses to find out whether the API is vulnerable to specific application-level attacks.

The difference is the starting point. A dynamic scan starts from a URL and crawls the site; an API scan starts from an **API specification** (a URL or a file). Instead of crawling, Syhunt maps the API from the specification and then runs a **schema validation** step, a form of functional testing that confirms the API exists and behaves as specified before the security tests begin. If it finds a major mismatch, the scan stops automatically and shows a warning, so you don't get misleading results.

Supported specification formats

- API Blueprint
- OpenAPI 2 and 3
- Swagger 1, 2 and 3
- Postman Collection 1, 2 and 2.1
- GraphQL (with REST annotations)
- I/O Docs
- Google Discovery
- RAML
- WADL

Before scanning, Syhunt checks that the specification URL is valid and in a supported format, and rejects anything incompatible or invalid. The basic license supports the main formats (OpenAPI 2 and 3, Swagger 1 to 3, and API Blueprint), which together cover more than 95% of the API specifications in use today. Other licenses may add GraphQL, I/O Docs, Google Discovery, RAML and WADL.

Because APIs are black boxes, Syhunt needs a specification to scan one: it provides the structure and endpoint definitions that guide testing. Syhunt is broadly compatible, but we recommend API Blueprint. Its clean, human-readable Markdown syntax, strong documentation features and easy integration into development and testing workflows fit Syhunt's focus on ease of use.

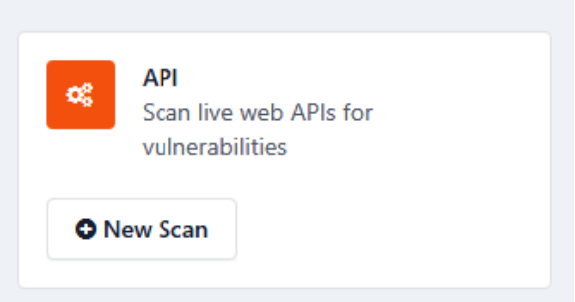
Running an API scan

1. ⚠️ Make sure you meet the pre-scan requirements and are authorized to scan the target API. Syhunt includes safeguards against scanning APIs outside your direct control.

Page 71 of 175

Syhunt Hybrid 7.2

2. Enable the Web UI, if you haven't done so yet.
3. In the Web UI, click API's **New Scan** button.



4. Enter the URL of the API's specification, for example `http://mysite/api/openapi.json`. To try it out, use one of the demo specifications.

> **Note**
> You'll get an **Invalid target** error if you enter the API's own URL instead of its specification URL, or if the specification is invalid.

5. Select a scan method. We recommend **Application Scan (Default)**, which checks for all vulnerabilities using the recommended settings. See Hunt Methods.
6. *(Optional)* Click **Preferences** next to the target URL to customize the API's settings.
7. Click **Start Scan**.

You can click **View Report** at any time during the scan to see the results so far as an HTML report.

To scan from the command line, see Command Line: API scans.

Demo specifications

The Web UI includes simulated vulnerable APIs for demonstration and testing. Point the scanner to one of the specification URLs below. These specs reference locally accessible endpoints only and contain no real vulnerabilities; they just simulate vulnerable behavior so you can test and learn safely.

Format	Specification URL
API Blueprint	`http://spec.syhunt.test/vulnapisim.apib`
OpenAPI 3	`http://spec.syhunt.test/vulnapisim_openapi3.json`
Swagger 2	`http://spec.syhunt.test/vulnapisim_swagger2.json`
Postman 2.1	`http://spec.syhunt.test/vulnapisim_postman2.1.json`
Postman 1	`http://spec.syhunt.test/vulnapisim_postman1.json`

Preventing a vulnerability from being reported

Findings from API scans can be suppressed the same way as in dynamic scans, using the ignore IDs shown in the reports.

Hunt methods for APIs

Most hunt methods available for dynamic scans also work for API scans, with one important difference in the **Top 10** method:

Page 72 of 175

Syhunt Hybrid 7.2

- For web applications, it covers the **OWASP Top 10 for Web Applications**: injection, broken authentication, cross-site scripting and so on.
- For APIs, it covers **both** the OWASP Top 10 for Web Applications **and** the **OWASP API Security Top 10**, which adds API-specific risks such as broken object-level authorization (BOLA), excessive data exposure, lack of rate limiting and API-specific misconfigurations.

This reflects a principle at Syhunt: the classic OWASP Top 10 is still highly relevant to APIs, because many web application risks also show up at the API level. Applying both lists gives a truly comprehensive Top 10 assessment, covering the risks APIs inherit from web applications as well as those unique to APIs.

Methods that don't apply to APIs are unavailable. If you start an API scan with one of them, Syhunt aborts and shows a warning.

Scheduling API scans

API scans are scheduled the same way as dynamic scans. See Scheduling Scans.

Pre-scan requirements

1. ⚠️ **An API specification (URL or file) is required.** Because APIs are black boxes, an accurate and complete scan is impossible without one: the specification defines the endpoints, parameters and expected behavior that guide testing.
    - Security teams don't always have access to API specifications. If that's your case, ask the development team for them before you start. Treat any API without proper documentation or a specification as a security risk, since it can't be effectively tested, reviewed or monitored.
2. ⚠️ **Only system administrators, or other people in charge of the target, should use this software. Do not scan APIs outside your direct control.**
    - If you need to scan an API you don't control, get written permission from its owner or administrator first.
    - Syhunt includes safeguards to prevent accidental scanning of APIs outside your organization's control.
    - As good practice, back up the API's source code and database before scanning. In rare cases, injection testing against an insecure, fragile API can pollute the database or alter files in ways that affect how the API works.
3. Make sure you meet the Internet connection requirements.
4. Read and accept the Syhunt EULA before launching any scan.

Hosting and uploading specifications

Since Syhunt 7.1.11, you can either link to a specification hosted anywhere or upload a specification file.

Source	How it works	Includes (referenced files)
Hosted	Enter the specification's URL.	Supported, as long as every included file is in the same path as the main specification, or below it.
Uploaded	Syhunt serves the file at an internal URL such as `http://spec.syhunt.test/<specfilename.ext>`.	Not supported. The file must be self-contained.
Local (Windows only)	Place the file in `C:\ProgramData\Syhunt\UserData\Specs`. It becomes available at `http://localspec.syhunt.test/<specfilename.ext>`, accessible only from the internal network to reduce exposure.	Supported. Syhunt fetches the included files automatically.

Safeguards against accidental scanning

Syhunt automatically aborts an API scan unless one of these conditions is met:

1. The API host referenced in the specification is on an internal network (a private IP range or a loopback address).
2. The API's domain has been explicitly allowed by adding it to the Targets list. For example, if the specification points to `api.mydomain.com`, add `mydomain.com` to the Targets list, either:

Page 73 of 175

Syhunt Hybrid 7.2

- in the Web UI at `http://127.0.0.1:8017/syhunt/?n=targets`, or
- from the CLI: `scancore -tg:https://mydomain.com -prefset:enabled -v:true`

If neither condition is met, an error message explains the problem.

This safeguard prevents users from unintentionally scanning external or third-party APIs that their organization doesn't control. It keeps scans within the authorized scope, protecting both the user and outside systems from unintended or unauthorized testing.

Specification format requirements

Syhunt makes a considerable effort to accept non-standard or loosely written specifications, but specifications that stray too far from their official format may be rejected, and Syhunt does not commit to supporting malformed or non-compliant specifications. If yours is rejected, correct it or use a third-party tool to make it compliant, or convert it to API Blueprint for the best compatibility and stability.

System requirements

See Technical Specifications.

Page 74 of 175

Hunt Methods

> **Applies to:** Syhunt Hybrid 7.0.11

A *hunt method* (scan method) decides what Syhunt looks for and how aggressively it tests. **Application Scan** is the default; choose another method when you need a narrower or deeper scan.

Comparison

Hunt method	CLI name	Type	Brute force	Injection	DoS	Time-consuming
Application Scan (default)	`appscan`	▣	Y	Y	Y	N
Application Scan (server-side focused)	`appscanss`	▣	Y	Y	Y	N
Structure Brute Force	`structbf`	■	Y (deep)	N	N	Y (very)
Old & Backup Files	`fileold`	■	Y	N	N	Y
Fault Injection	`faultinj`	▣	N	Y	Y	N
OWASP Top 10	`top10`	▣	N	P (Top 10)	Y	N
OWASP Mobile Top 10	`top10mob`	□	N	P (Mobile Top 10)	N	N
CWE Top 25	`top25cwe`	▣	N	P (Top 25)	Y	N
OWASP PHP Top 5	`top5php`	▣	N	P (Top 5)	N	N
Cross-Site Scripting	`xss`	▣	N	P (XSS)	N	N
SQL Injection	`sqlinj`	▣	N	P (SQL)	N	N
File Inclusion	`fileinc`	▣	N	P (FI)	N	N
Unvalidated Redirects	`unvredir`	▣	N	P (UR)	N	N
Malware Content	`malscan`	▣	P (malware)	P (malware)	N	N
Passive	`passive`	▣	N	N	N	N
Spider Only	`spider`	■	N	N	N	N
Complete Scan	`complete`	▣	Y	Y	Y	Y (very)
Complete Scan, No DoS	`compnodos`	▣	Y	Y	N	Y (very)
Complete Scan, Paranoid	`comppnoid`	▣	Y (deep)	Y	Y	Y (very)

Y = yes, **N** = no, **P** = partial (limited to the category shown).

Type of testing

- ▣ Hybrid (gray box): dynamic and code
- ■ Dynamic only (black box)
- □ Code only (white box)

Time-consuming: *Y* means the method runs extra checks and attack mutations, and the number of checks grows with the number of directories found while crawling.

Method descriptions

Syhunt Hybrid 7.2

Application Scan

Finds flaws in custom web applications, web server software and third-party components. It crawls the site and attacks both the site structure and the applications, including fault injection tests such as XSS, SQL injection and file inclusion.

Structure Brute Force

Checks for:

- Common vulnerable scripts
- Common files
- Custom (user-defined) files
- Database disclosure
- Web-based backdoors

The number of checks grows with the number of directories found while crawling.

Old & Backup Files

Tries backup and old-file extensions against every item in the mapped site structure.

OWASP Top 10

Scans specifically for the OWASP Top 10 (2017) risks:

1. A1: Injection
2. A2: Broken Authentication
3. A3: Sensitive Data Exposure
4. A4: XML External Entities (XXE)
5. A5: Broken Access Control
6. A6: Security Misconfiguration
7. A7: Cross-Site Scripting (XSS)
8. A8: Insecure Deserialization
9. A9: Using Components with Known Vulnerabilities
10. A10: Insufficient Logging & Monitoring

CWE Top 25

Scans specifically for the 2019 CWE Top 25 Most Dangerous Software Errors.

OWASP PHP Top 5

Scans specifically for the OWASP Top 5 PHP vulnerabilities:

1. Remote command execution
2. Cross-site scripting (XSS), including DOM XSS
3. SQL injection
4. PHP misconfiguration
5. File system attacks, including file inclusion

Fault Injection

Scans only for fault injection vulnerabilities. All checks that don't involve injection are disabled, and Syhunt focuses on SQL injection, XSS, file inclusion and similar flaws.

Cross-Site Scripting (XSS)

Scans specifically for XSS vulnerabilities, including DOM-based XSS.

Page 76 of 175

Syhunt Hybrid 7.2

SQL Injection

Scans specifically for SQL and NoSQL injection vulnerabilities.

File Inclusion

Scans specifically for file inclusion and directory traversal vulnerabilities.

Unvalidated Redirects

Scans specifically for unvalidated redirect vulnerabilities.

Malware Content

Scans specifically for malicious content, such as:

- Web backdoors
- Malicious content
- Hidden debug parameters

Passive

Maps the site structure and reports issues without launching any attacks, for example:

- Vulnerabilities in client-side JavaScript
- Form weaknesses
- Web technology disclosure
- Insecure HTTP headers
- Outdated, vulnerable server software
- Outdated, vulnerable referenced scripts
- Suspicious HTML comments
- Source code disclosure
- Malicious content being served

Spider Only

Maps the site structure without testing for or reporting any vulnerability or weakness.

Complete Scan

Tests for every kind of web application vulnerability using all available mutations and penetration-testing techniques, including header manipulation attacks. It can take a long time against servers with many folders and entry points.

Complete Scan, No DoS

Same as Complete Scan, with denial-of-service tests disabled.

Complete Scan, Paranoid

Same as Complete Scan, plus a deep structure brute force. It can take a very long time, especially on large sites. It also runs a triple-check brute force for case-sensitive servers, trying every variation of each file name (all uppercase, all lowercase, leading capitals and so on).

Page 77 of 175

Syhunt Hybrid 7.2

Ignore Rules

> **Applies to:** Syhunt Hybrid 7.2

Ignore rules stop matching vulnerabilities from being reported. They complement ignore IDs (shown at the end of each vulnerability entry in a report) when you need to suppress a broader set of findings. You can add them:

- per target, in the target's exclusion list (see Dynamic Scanning);
- per repository, in a `.vulnignore` file (see Source Code Scanning);
- globally, for all scans (see Global rules below).

Syntax

A rule starts with `rule:` followed by comma-separated `parameter=value` pairs. Wrap a pair in double quotes if its value contains spaces or commas.

Examples

Rule	Effect
`rule:path=*,checkid=C-1603659845-9557`	Ignores any vulnerability with check ID `C-1603659845-9557`.
`rule:path=*,name=XSS`	Ignores any vulnerability with "XSS" in its title.
`rule:path=/demo/*,name=XSS`	Ignores vulnerabilities with "XSS" in the title whose path starts with `/demo/`.
`rule:path=*,"name=Web Technology Disclosure"`	Ignores any vulnerability with "Web Technology Disclosure" in its title.
`rule:path=/demo.php,lines=10`	*(SAST)* Ignores vulnerabilities on line 10 of `/demo.php`.
`rule:path=/demo.php,"lines=10,11"`	*(SAST)* Ignores vulnerabilities on lines 10 and 11 of `/demo.php`.

Parameters

Parameter	Matches against
`path` **(required)**	The affected path. Accepts the wildcards `?` and `*`.
`checkid`	The vulnerability's check ID.
`name`	The vulnerability's title.
`params`	The affected parameter(s). Separate multiple parameters with commas.
`risk`	The vulnerability's risk: `low`, `medium`, `high` or `info`.
`module`	The module that found the vulnerability: `dyn` or `code`. If omitted, the rule applies to both Dynamic and Code findings.
`lines`	The affected source code line(s). Separate multiple lines with commas.
`cve`	A CVE ID among the vulnerability's CVE references.
`cwe`	A CWE number among the vulnerability's CWE references.

Global rules

To create rules that apply to every dynamic or code scan:

Page 78 of 175

Syhunt Hybrid 7.2

1. Open the Connectors screen (☰ **Menu** → **Connectors**) and add a connector of type **List** named `VULNIGNORE-DAST` (for dynamic scans) or `VULNIGNORE-SAST` (for code scans).
2. Click the connector in the list to add or edit its rules.

Page 79 of 175

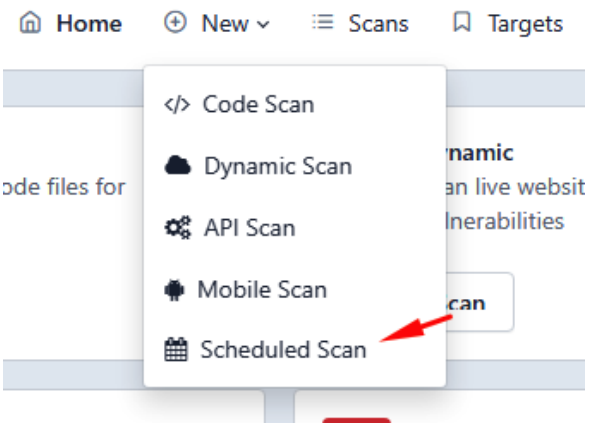
Syhunt Hybrid 7.2

Scheduling Scans

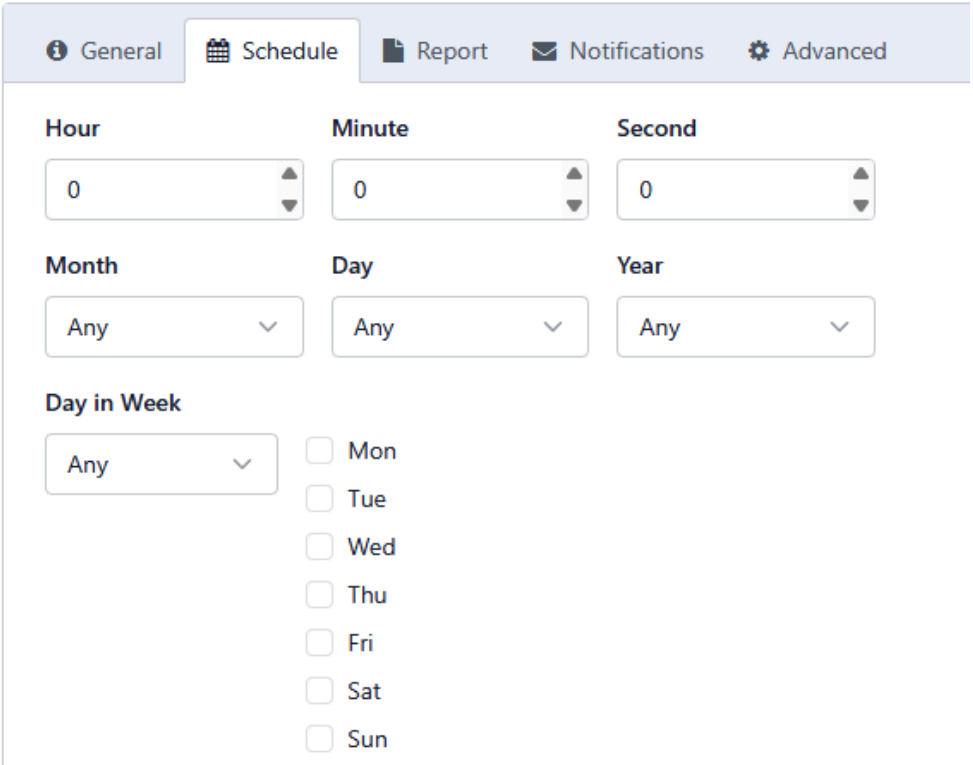
> **Applies to:** Syhunt Hybrid 7.2

How to schedule a scan

1. Enable the Web UI, if you haven't done so yet.
2. In the Syhunt Web UI, click **New** → **Scheduled Scan**.



3. The **Schedule New** screen opens.
4. On the **General** tab, enter a name for the scheduled scan (for example, `MyScan`), fill in the target details and choose a scan method.
5. On the **Report** tab, set the report generation options.
6. On the **Schedule** tab, define when the scan should run.



7. Click **Save**.
8. Confirm that the new schedule appears in the list of scheduled scans at `https://127.0.0.1:8018/syhunt/?n=scanschedule`.

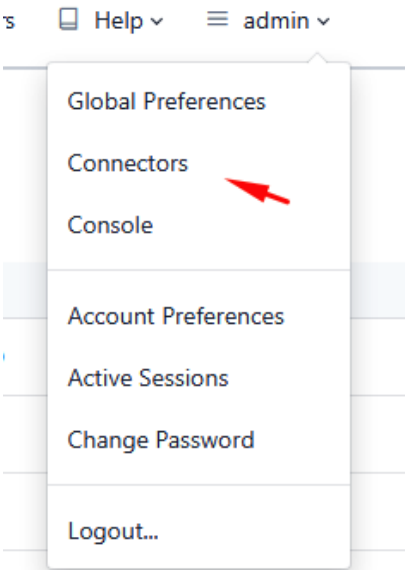
Page 80 of 175

Syhunt Hybrid 7.2

Emailing reports

1. Create an Email connector

1. Click **Connectors** in the top menu of the Web UI.



2. Click **Add**. The **New Connector** screen opens.
3. Enter a name for the connector (for example, `MyEmail`).
4. Set **Type** to **Email**.
5. Click **Create** and, after the success message, **Close**.

2. Configure the connector

1. Click the connector you just created. Its preferences screen opens.
2. Enter the sender and recipient email addresses.
3. Enter the SMTP host and authentication credentials.
4. Click **Save**.

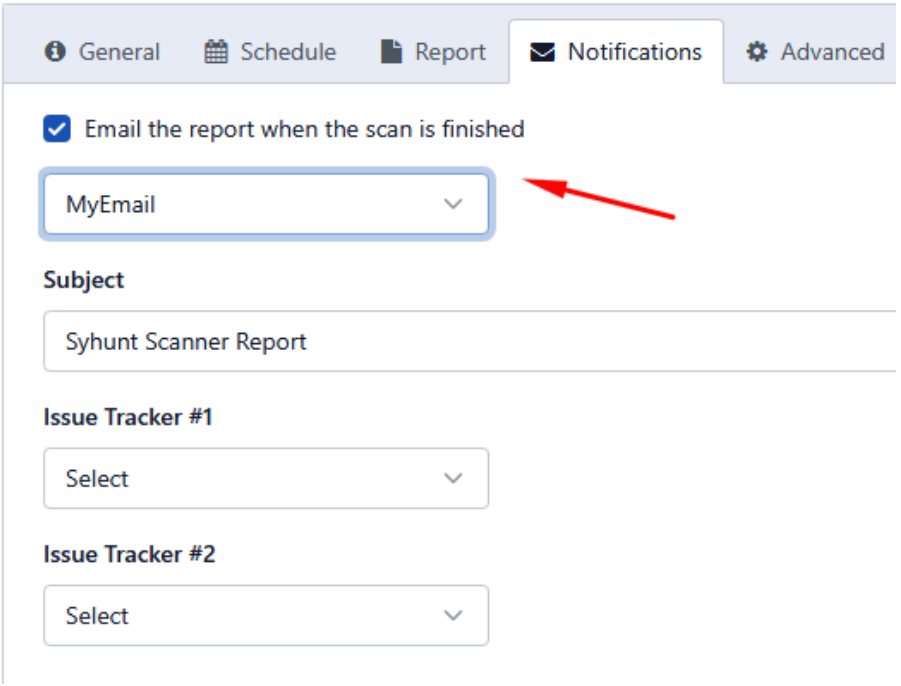
3. Attach the connector to a scheduled scan

1. Open **Scans** from the top menu and click **Scheduled Scans**.
2. Click the scheduled scan whose reports you want emailed. Its preferences dialog opens.

Page 81 of 175

Syhunt Hybrid 7.2

3. On the **Notifications** tab, select **Email the report when the scan is finished**.



4. Choose your email connector from the list (`MyEmail` in the screenshot above).
5. Click **Save**.

Reviewing results

You can review the results of past and running scans at any time: open the Syhunt Hybrid home page and click **Scans** in the top menu.

Page 82 of 175

Syhunt Hybrid 7.2

Forensic Log Analysis

> **Applies to:** Syhunt Forensic 7.0.13

> **Important**
> In Syhunt Hybrid 7.2, Forensic is available only from the command line (the `scanlog` command; see the Command Line guide). A future update will make it available from the Web UI again. The interface steps below apply to earlier versions.

Introduction

Syhunt Forensic (formerly Syhunt Insight) investigates web application compromises and helps prevent future breaches. It quickly and accurately identifies the source, type and methods of an attack, so organizations can close the gaps and defend against further threats.

Syhunt Forensic is not a standalone product. It complements Syhunt's dynamic (DAST) and static (SAST) testing tools by focusing on post-incident analysis and breach investigation: while DAST and SAST find and fix vulnerabilities during development and deployment, Forensic shows in detail what happened in attacks that have already taken place. Together they cover both prevention and response.

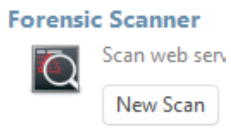
Supported log formats

Syhunt Forensic reconstructs attack sessions from the logs of popular web servers. It relies on the details these logs record to tell legitimate traffic from malicious traffic, and automated attacks from manual ones, so it needs access to them:

- Apache
- Nginx
- Microsoft IIS
- Lighttpd

Running a forensic scan

1. Launch Syhunt Hybrid and click the Syhunt Forensic icon, or **New Scan**, on the welcome page.



2. Select the web server log file to analyze. This is usually a file with a `.log` extension, such as `access.log`, or a file whose name starts with `access_log` and has no extension.
3. Wait for the scan to finish.

When the scan ends, the log gets one of these statuses:

Status	Meaning
No attacks	No application security attacks were found in the log.
Attack Attempts Found	There are signs of possible attacks, but no confirmed breach.
Breached	There are signs of possible attacks **and** of a successful breach.

Click **Save Results** in the toolbar to save the list of attempted attacks.

Page 83 of 175

Syhunt Hybrid 7.2

Reconstructing an attacker's session

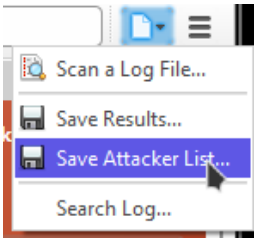
Syhunt Forensic can reconstruct entire attack sessions from the log, separating legitimate from malicious traffic and automated from manual attacks. This helps you understand the full scope of an incident.

1. Double-click an attack in the list.
2. In the bottom bar, click **View All Requests**.
3. A new tab opens. Wait for the reconstruction to finish.

Exporting the attacker list

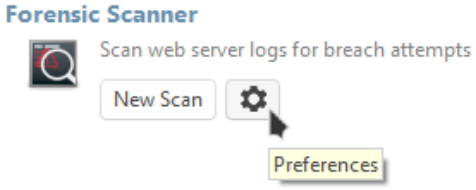
You can export the IP addresses identified as attackers and use the list to harden your firewall rules, blocking those addresses proactively to prevent further attacks from the same sources.

When the scan has finished, click the file icon shown below and choose **Save Attacker List**.



Excluding IP addresses from the analysis

1. Click **Preferences**.



2. Go to the **Exclusions** tab.
3. Click **Source IPs**.
4. Add the IP addresses to ignore, or import them from a file.
5. Click **OK** to save the list, then **OK** again to close the preferences dialog.
6. Run the scan again.

Page 84 of 175

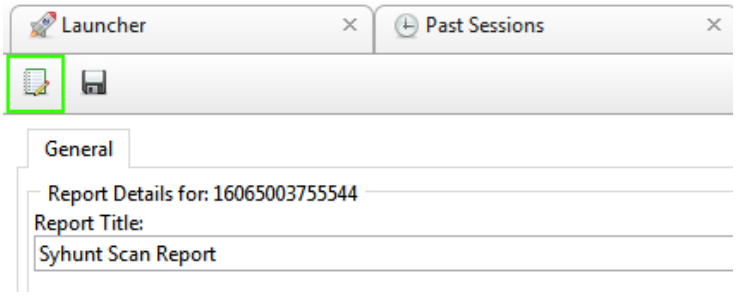
Reporting

> **Applies to:** Syhunt Hybrid 6.9

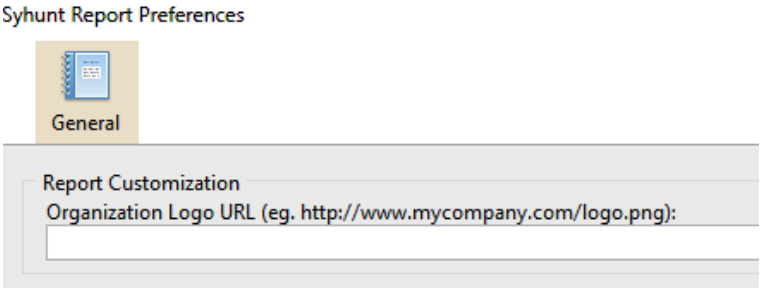
Customizing reports

Before saving a report, you can choose its language and add your organization's logo. The settings apply to every report you generate from then on.

1. Click **Edit Report Preferences** in the toolbar. The **Report Preferences** dialog opens.



2. Enter the URL of the image that contains your logo.



3. Choose the output language.
4. Click **OK** to save.

Reports you generate from now on will show your organization's logo instead of Syhunt's.

Exporting results

When a scan finishes, click **Export** on the scan results screen to download the results in your preferred format. For the full list of report and export formats, see Technical Specifications.

Syhunt Hybrid 7.2

Integrations

Page 86 of 175

Syhunt Hybrid 7.2

Command Line (CLI)

> **Applies to:** Syhunt Hybrid 7.2

Contents

- Introduction
- Dynamic scans
- API scans
- Source code scans
- Forensic scans
- Pass/fail conditions
- Incremental scan modes
- Report formats, templates and languages
- Modifying Syhunt preferences
- Using issue trackers
- Updating Syhunt

Introduction

This guide explains how to run scans and generate vulnerability reports from the command line.

The command-line tools are in the Syhunt installation folder, which depends on your operating system and edition:

OS	Default location

Windows	`C:\Program Files\Syhunt Hybrid\`
	`C:\Program Files (x86)\Syhunt Hybrid\`
	`C:\Program Files\Syhunt Community`

Linux	`/home/<user>/syhunt-hybrid/carbon/` `/home/<user>/syhunt-community/carbon/`
macOS	`/Applications/Syhunt Hybrid/carbon` `/Applications/Syhunt Community/carbon`

Tool	Purpose
`scanurl`	Dynamic (DAST) and API scans
`scancode`	Source code scans (SAST/MAST)
`scanlog`	Forensic log analysis
`scancore`	Preferences, trackers and maintenance commands
`scanupdate`	Updates (Linux and macOS)

Open a terminal or Command Prompt in the installation folder before running the commands below.

Dynamic scans

```
scanurl <starturl> -hm:<huntmethod>
```

Example:

Page 87 of 175

Syhunt Hybrid 7.2

```
scanurl http://www.somehost.com -hm:appscan
```

`scanurl` automatically generates and saves a report unless you pass `-nr`.

`scanurl` parameters

All parameters are optional.

Parameter	Description	Default
`-sn:<name>`	Session name; must be unique. If omitted, a unique ID is generated.	Generated ID
`-hm:<name>`	Hunt method to use.	`appscan`
`-emu:<mode>`	Browser emulation mode: `chrome`, `edge`, `firefox`, `msie` or `safari`.	`chrome`
`-srcdir:<dir>`	Target code folder for a hybrid scan (for example `"C:\www\docs\"` or `/home/user/www/`).	
`-tk:<tracker>`	Sends vulnerabilities to a tracker after the scan. Can be combined with `-pfcond`.	
`-tk2:<tracker>`, `-tk3:<tracker>`	Additional trackers, as above.	
`-nr`	Doesn't generate a report after the scan.	
`-or`	Opens the report after generating it.	
`-rout:<filename>`	Report file name; the extension sets the format.	`Report_<session>.html`
`-rtpl:<name>`	Report template.	`Standard`
`-xout:<filename>`	Export file name; the extension sets the format.	`Export_<session>.xml`
`-xout2:<filename>`	A second export file.	`Export_<session>.xml`
`-pfcond:<condition>`	Pass/fail condition to report.	
`-nv`	Less verbose output (errors and basic information are still printed).	
`-inc:<mode>`	Incremental scan mode.	`targetpref`
`-inctag:<name>`	Stores the incremental scan data under a tag.	
`-mnl:<n>`	Maximum number of links per server.	`10000`
`-mnr:<n>`	Maximum number of retries.	`2`
`-tmo:<ms>`	Request timeout, in milliseconds.	`8000`
`-tml:<time>`	Maximum scan time, for example `1d`, `3h`, `2h30m`, `50m`.	No limit
`-ver:<v>`	HTTP version.	`1.1`
`-nofris`	Doesn't automatically follow an off-domain redirect at the start URL.	
`-nodos`	Disables denial-of-service tests.	
`-nojs`	Disables JavaScript emulation and execution.	
`-atype:<type>`	Authentication type: `Basic`, `Form` or `Manual`.	
`-auser:<username>`	Username for authentication.	
`-apass:<password>`	Password for authentication.	

Page 88 of 175

Syhunt Hybrid 7.2

Parameter	Description	Default
`-about`	Shows version information.	
`-help` (or `/?`)	Lists the available parameters.	

Scanning IPv6 addresses

Enclose IPv6 addresses in square brackets, for example `http://[2001:4860:0:2001::68]/index.php`.

API scans

API scans use `scanurl` too. The only difference is that you pass the URL of the **API specification** instead of a start URL. If you pass the API's own URL, Syhunt runs a traditional dynamic scan instead, without API mapping, schema validation or API-specific tests.

For example, if the specification is at `https://example.com/openapi.json` and describes an API at `https://api.example.com`:

```
scanurl https://example.com/openapi.json -hm:appscan   # correct: API scan
scanurl https://api.example.com                        # wrong: runs a traditional DAST scan
```

`scanurl` accepts the same optional parameters for API scans. See Web API Scanning for details.

Source code scans

```
scancode <target> -hm:<huntmethod>
```

Examples:

```
:: Git repositories
scancode git://sub.domain.com/repo.git
scancode https://github.com/user/repo.git -rb:master

:: Local paths
scancode /source/www/
scancode c:\source\www\
scancode c:\source\www\file.php
scancode c:\mobile\myapp.apk
scancode "c:\source code\www\"
```

Azure DevOps / TFS repositories:

```
scancode https://dev.azure.com/user/project
scancode https://myserver/tfs/project
scancode collection:https://dev.azure.com/user$/project
```

`scancode` automatically generates and saves a report unless you pass `-nr`.

`scancode` parameters

All parameters are optional.

Parameter	Description	Default
`-sn:<name>`	Session name; must be unique. If omitted, a unique ID is generated.	Generated ID
`-hm:<name>`	Hunt method to use.	`appscan`
`-rb:<branch>`	Git repository branch.	

Page 89 of 175

Parameter	Description	Default
`-tfsv:<version>`	TFS version.	`default`
`-tk:<tracker>`	Sends vulnerabilities to a tracker after the scan. Can be combined with `-pfcond`.	
`-tk2:<tracker>`, `-tk3:<tracker>`	Additional trackers, as above.	
`-nr`	Doesn't generate a report after the scan.	
`-or`	Opens the report after generating it.	
`-rout:<filename>`	Report file name; the extension sets the format.	`Report_<session>.html`
`-rtpl:<name>`	Report template.	`Standard`
`-xout:<filename>`	Export file name; the extension sets the format.	`Export_<session>.xml`
`-xout2:<filename>`	A second export file.	`Export_<session>.xml`
`-pfcond:<condition>`	Pass/fail condition to report.	
`-nv`	Less verbose output (errors and basic information are still printed).	
`-inc:<mode>`	Incremental scan mode.	`targetpref`
`-inctag:<name>`	Stores the incremental scan data under a tag.	
`-excp:<paths>`	Paths to exclude from the analysis, for example `-excp:/path/*,/path2/*`.	
`-refurl:<url>`	A URL associated with the code, for reference only.	
`-noifa`	Disables input filtering analysis.	
`-tml:<time>`	Maximum scan time, for example `1d`, `3h`, `2h30m`, `50m`.	No limit
`-about`	Shows version information.	
`-help` (or `/?`)	Lists the available parameters.	

Forensic scans

Forensic log analysis is run with `scanlog`. In Syhunt Hybrid 7.2, this is the only way to run Syhunt Forensic. See Forensic Log Analysis. Run `scanlog -help` to list its parameters.

Pass/fail conditions

Pass a pass/fail condition to `scanurl` or `scancode` with the `-pfcond` parameter.

Condition	Fails if...
`high` or `fail-if:risk=high`	a High-risk vulnerability or threat is found.
`medium` or `fail-if:risk=mediumup`	a Medium- or High-risk vulnerability or threat is found.
`low` or `fail-if:risk=lowup`	a Low-, Medium- or High-risk vulnerability or threat is found.

Incremental scan modes

Mode	Description
`auto`	Manages the incremental scan cache automatically (recommended).
`disabled`	Disables the incremental scan cache. Scans take 3 to 4 times longer.
`forced`	Always uses incremental scanning. If you use this mode, also run a regular, non-forced scan about once a month.
`targetpref`	Uses the mode set in the target's preferences.

To reset the incremental scan cache, run `scancore -runcmd:clearinc`.

Report formats, templates and languages

Languages

Code	Language
`EN`	English
`DE`	German
`ES`	Spanish
`FR`	French
`IT`	Italian
`JP`	Japanese
`KR`	Korean
`PT`	Portuguese

To set the report language:

```
scancore -prefset:hybrid.report.defaultlanguage -v:EN
```

You can also set it in the interface: ☰ **Menu** → **Preferences** → **Other Hybrid Preferences**.

Formats

The output format is set by the file extension passed to `-rout` or `-xout`:

Extension	Format
`.html`	HTML report
`.pdf`	PDF report
`.xml`	XML export
`.json`	JSON vulnerability export
`.csv`	CSV (comma-separated values)
`.mse.csv`	CSV for Microsoft Excel
`.txt`	Text report

Syhunt Hybrid 7.2

Templates

Template	Contents
`Standard`	Low- to high-risk vulnerability information.
`Comparison`	Standard content plus how the vulnerabilities evolved between scans.
`Compliance`	OWASP Top 10, CWE/SANS Top 25 (2019) and PCI DSS 3.2.1 compliance information.
`Mobile`	OWASP Mobile Top 10, CWE/SANS Top 25 (2019) and PCI DSS 3.2.1 compliance information.
`Complete`	Standard content plus comparison, compliance, request/response and coverage details.

For the available hunt methods and their CLI names, see Hunt Methods.

Modifying Syhunt preferences

Since Syhunt 6.9.6.3, you can change Syhunt's persistent preferences, globally or for a specific target, with `scancore -prefset`. Use `-prefprint` to print the current value of any preference.

```
-- Example 1: set the logo shown in new reports
scancore -prefset:hybrid.report.company.logo.urldark -v:https://www.mydomain.com/mylogo.png

-- Example 2: print the current report logo URL
scancore -prefprint:hybrid.report.company.logo.urldark

-- Example 3: set Basic authentication for a specific site
scancore -tg:http://127.0.0.1 -prefset:dynamic.formauth.type -v:Basic
scancore -tg:http://127.0.0.1 -prefset:dynamic.formauth.username -v:myuser
scancore -tg:http://127.0.0.1 -prefset:dynamic.formauth.password -vsecret

-- Example 4: enable or disable the preferences of a specific site
scancore -tg:http://127.0.0.1 -prefset:enabled -v:true
scancore -tg:http://127.0.0.1 -prefset:enabled -v:false

-- Example 5: set a value from a list file
scancore -tg:http://127.0.0.1 -prefset:dynamic.lists.cookies -fromfile:mycookies.lst
```

See Preferences for the full list of preference IDs.

Clearing all preferences

```
-- Clear global preferences
scancore -runcmd:clearpref
-- Clear the preferences of all sites
scancore -runcmd:clearsite
```

Using issue trackers

Since Syhunt 6.9.6.3, you can add, manage and use issue trackers from the command line with `scancore -tracker`.

```
-- Example 1: add a GitHub tracker
scancore -tracker:add
-- Enter the type (GitHub) and a name for the tracker, then press Enter.
-- Configure it:
scancore -tracker:set to:mytrackername -key:project.name -v:owner/repo
scancore -tracker:set to:mytrackername -key:auth.token.encrypted -vsecret

-- Example 2: add an email tracker
scancore -tracker:add
-- Enter the type (Email) and a name for the tracker, then press Enter.
scancore -tracker:set to:myemailtracker -key:message.from -v:robot@yourdomain.com
scancore -tracker:set to:myemailtracker -key:message.tolist -v:security@yourdomain.com,team@yourdomain.com
scancore -tracker:set to:myemailtracker -key:auth.targethost -v:smtp.yourdomain.com
scancore -tracker:set to:myemailtracker -key:smtp.targetport -v:587
```


Page 92 of 175

Syhunt Hybrid 7.2

```
scancore -tracker:set to:myemailtracker -key:auth.username -v:myusername
scancore -tracker:set to:myemailtracker -key:auth.password -vsecret

-- Example 3: test a tracker
scancore -tracker:send -tid:TEST -to:mytrackername -note:"My comment"

-- Example 4: send a vulnerability from a report to a tracker
-- 1596281007-7-4771 is the vulnerability's track ID, taken from the report.
scancore -tracker:send -tid:1596281007-7-4771 -to:mytrackername -note:"My comment"

-- Example 5: list trackers
scancore -tracker:list

-- Example 6: delete a tracker
scancore -tracker:del to:mytrackername
```

See Preferences: trackers for the list of tracker preference IDs.

Sending vulnerabilities to a tracker after a scan

Pass `-tk` to `scanurl` or `scancode` to send a summary of the vulnerabilities found to a tracker:

```
-- Example 1: use an existing tracker
scanurl mydomain -tk:trackername

-- Example 2: send only if high-severity vulnerabilities are found
scanurl mydomain -tk:trackername -pfcond:fail-if:risk=high

-- Example 3: define the tracker on the fly
scanurl mydomain -tk:"?app=github###project=user/repo###token=ghp_etc"
```

> **Note**
> When you pass tracker preferences in `-tk`, use the short versions of their names.

Removing all trackers

```
scancore -runcmd:cleartrack
```

Updating Syhunt

On Linux and macOS, run `scanupdate` to check for updates; if any are available, Syhunt asks whether to download and install them. Run `scanupdate auto` to install them without confirmation.

On Windows, download and install updates from the Syhunt website.

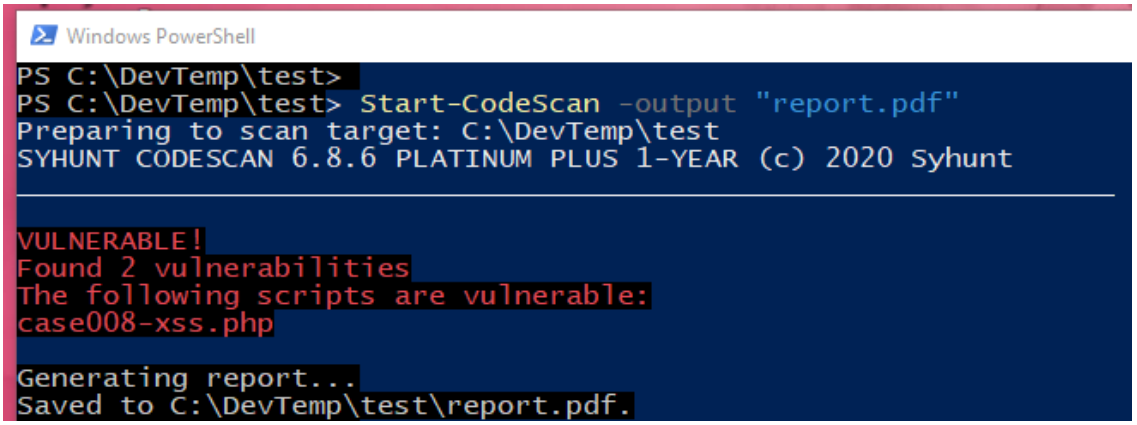
Page 93 of 175

Syhunt Hybrid 7.2

PowerShell

> **Applies to:** Syhunt Hybrid 6.9.17

You can run Syhunt scans from PowerShell scripts, which makes it easy to bring Syhunt Dynamic, Syhunt Code and Syhunt Mobile into your continuous delivery pipeline and other automation. The Syhunt Hybrid installer adds a **Syhunt PowerShell module** on Windows, with functions to start scans, retrieve reports and results, and perform pass/fail testing.



Adding Syhunt to a script

Insert code like this at the appropriate point in your PowerShell script:

```
# SAST: scan the current directory/repository
Start-CodeScan -pfcond "medium"

# SAST: scan a remote repository
$MyProject = @{
  target = 'https://github.com/syhunt/vulnphp.git';
  branch = 'main';
  pfcond = 'medium';
  output = 'report.pdf'
}
Start-CodeScan @MyProject

# DAST: scan a URL
$MyWebsite = @{
  target = 'https://www.somewebsite.com';
  pfcond = 'medium';
  output = 'report.pdf'
}
Start-DynamicScan @MyWebsite
```

`Start-DynamicScan`

Launches a Syhunt Dynamic scan.

Parameter	Description
`target` **(required)**	The URL to scan, for example `http://www.somesite.com`.
`huntmethod`	The hunt method to use. If omitted, the default method is used.
`pfcond`	Makes the script exit with a failure code when a condition is met. See Pass/fail conditions.

Page 94 of 175

Syhunt Hybrid 7.2

Parameter	Description
`tracker`	The name of an existing tracker, or a tracker defined on the fly, that will receive a summary of the vulnerabilities found. See examples.
`output`	Output file name, for example `report.pdf` or `report.html`.
`outputex`	A second output file name, for example `export.json`.
`verbmode`	`$false` by default. Set to `$true` for verbose output (beyond errors and basic information).
`genrep`	`$true` by default. Set to `$false` to skip generating an output file.
`redirIO`	`$true` by default. Set to `$false` to stop redirecting the scanner's input and output to the console.
`timelimit`	Maximum scan time, for example `1d`, `3h`, `2h30m`, `50m`. The scan is aborted when the limit is reached. No limit by default.

`output` and `outputex` accept every output format Syhunt supports. Files are saved to the current working directory unless you give a full path.

```
# Example 1: one line
Start-DynamicScan -target 'https://www.somewebsite.com' -pfcond 'fail-if:risk=mediumup'

# Example 2: parameters in a hashtable
$MyWebsite = @{
  target = 'https://www.somewebsite.com';
  pfcond = 'medium';
  output = 'report.pdf'
}
Start-DynamicScan @MyWebsite
```

`Start-CodeScan`

Launches a Syhunt Code scan. All parameters are optional.

Parameter	Description
`target`	The URL of a repository, or a local source code folder or file. If omitted, the current working directory is scanned.
`branch`	The branch to scan. If omitted, the Git client fetches the default branch.
`huntmethod`	The hunt method to use. If omitted, the default method is used.
`pfcond`	Makes the script exit with a failure code when a condition is met. See Pass/fail conditions.
`tracker`	The name of an existing tracker, or a tracker defined on the fly, that will receive a summary of the vulnerabilities found. See examples.
`output`	Output file name, for example `report.pdf` or `report.html`.
`outputex`	A second output file name, for example `export.json`.
`verbmode`	`$false` by default. Set to `$true` for verbose output.
`genrep`	`$true` by default. Set to `$false` to skip generating an output file.
`redirIO`	`$true` by default. Set to `$false` to stop redirecting the scanner's input and output to the console.
`timelimit`	Maximum scan time, for example `1d`, `3h`, `2h30m`, `50m`. No limit by default.

```
# Example 1: scan the current directory/repository
Start-CodeScan -pfcond "medium"

# Example 2: scan a remote Git repository
Start-CodeScan -target "https://github.com/someuser/somerepo.git" -huntmethod "normal" -pfcond "medium"
```


Page 95 of 175

Syhunt Hybrid 7.2

```
# Example 3: scan an Azure DevOps Services project
Start-CodeScan -target "https://dev.azure.com/user/projectname" -huntmethod "normal" -pfcond "medium"

# Example 4: scan a specific local folder
Start-CodeScan -target "C:\www\" -huntmethod "normal" -pfcond "medium"
```

Pass/fail conditions

Condition	Fails if...
`high` or `fail-if:risk=high`	a High-risk vulnerability or threat is found.
`medium` or `fail-if:risk=mediumup`	a Medium- or High-risk vulnerability or threat is found.
`low` or `fail-if:risk=lowup`	a Low-, Medium- or High-risk vulnerability or threat is found.

Troubleshooting

"Script.ps1 cannot be loaded because running scripts is disabled on this system"

Start PowerShell as an administrator and allow scripts to run:

```
Set-ExecutionPolicy RemoteSigned
```

"The term 'Start-CodeScan' is not recognized..."

The Syhunt module isn't loaded. Run `Import-Module Syhunt` and call the function again.

Loading the module automatically

On Windows 10 and later, the Syhunt module loads automatically.

On Windows 7, either run `Import-Module Syhunt` before calling the Syhunt functions, or add it to your PowerShell profile so it loads every time:

1. Go to your PowerShell profile folder. By default the profile is:

   ```
   C:\Documents and Settings\User\My Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1
   ```

2. Create the `Microsoft.PowerShell_profile.ps1` file.
3. Add the line `Import-Module Syhunt` and save the file.

Page 96 of 175

Syhunt Hybrid 7.2

Web API

> **Applies to:** Syhunt Hybrid 6.9.8.2

Contents

- Introduction
- Launching a dynamic scan
- Launching a source code scan
- Getting results

Introduction

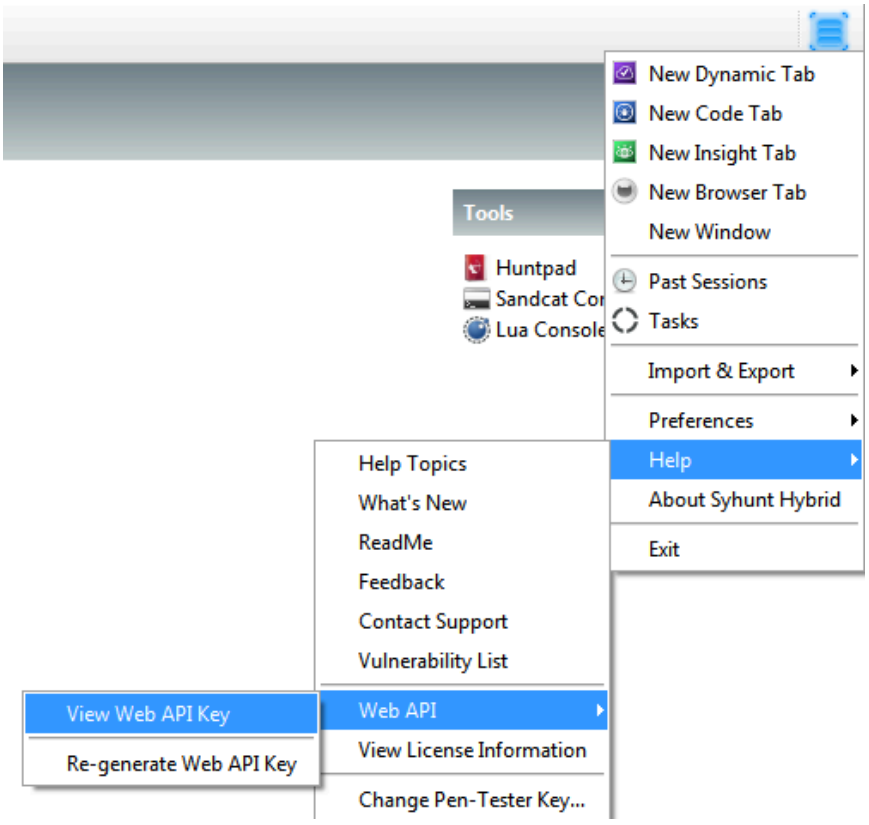
Syhunt Hybrid Platinum includes a simple web API for launching dynamic and source code scans and retrieving the status, report and log of a scan session. Requests are `POST` requests with a JSON body, and responses are JSON, except when you request a report, which is returned as XML or JSON.

The same operations are available from Lua; see the Lua API.

Before you start

Make sure the Syhunt API server is running and that you have an API key.

- **Linux:** start the API server with `scancore -apisignal:start` and generate an API key with `scancore -apikeygen`.
- **Windows:** in the Syhunt installation folder (usually `C:\Program Files\Syhunt Hybrid\`), open the `Server` subfolder and double-click the `nginx` executable to start the server. To get an API key, launch Syhunt Hybrid and choose **Help → View Web API Key**.



Then open `http://localhost:8017/` and check that you see the NGINX default page. The API is served at:

Page 97 of 175

Syhunt Hybrid 7.2

- `http://<hostname>:8017/syhunt/launch.lua`
- `http://<hostname>:8017/syhunt/results.lua`

Every request to these endpoints must include a valid API key in its body (`apikey`).

Launching a dynamic scan

Endpoint: `POST /syhunt/launch.lua`

```
{
  "sessionname": "Test",
  "starturl": "http://127.0.0.1",
  "huntmethod": "appscan",
  "reporttemplate": "Complete",
  "apikey": "YOUR_API_KEY"
}
```

Key	Description
`starturl` **(required)**	The target URL.
`apikey` **(required)**	Your Syhunt Web API key (see Before you start).
`sessionname`	A unique, alphanumeric session name. If omitted, a random one is generated.
`huntmethod`	A valid hunt method name.
`reporttemplate`	A report template: `Standard` , `Comparison` , `Compliance` or `Complete` .
`reportgl`	If `true` , returns JSON output in GitLab format.
`timelimit`	Maximum scan time, for example `1d` , `3h` , `2h30m` , `50m` . The scan is aborted when it's reached. No limit by default.

Response:

```
{
  "huntmethod": "appscan",
  "pid": 5108,
  "result": true,
  "resultstr": "",
  "sessionname": "Test",
  "sessiontype": "dynamic"
}
```

Key	Description
`pid`	A unique process ID for the running scan.
`result`	`true` if the scan was launched; `false` if it couldn't be started.
`resultstr`	If `result` is `false` , a description of the error.
`sessionname`	The scan's session name. If you didn't provide one, this is the generated name. Use it in later requests to get the results.
`sessiontype`	The session type: `dynamic` or `code` .

Launching a source code scan

Endpoint: `POST /syhunt/launch.lua`

Local folder:

```
{
  "sessionname": "Test",
```


Page 98 of 175

Syhunt Hybrid 7.2

```
  "sourcetarget": "P:\\Private\\MyWebApp\\",
  "huntmethod": "normal",
  "reporttemplate": "Complete",
  "apikey": "YOUR_API_KEY"
}
```

Repository URL:

```
{
  "sessionname": "Test",
  "sourcetarget": "https://github.com/drnic/php-helloworld.git",
  "sourcebranch": "master",
  "huntmethod": "normal",
  "reporttemplate": "Complete",
  "apikey": "YOUR_API_KEY"
}
```

Key	Description
`sourcetarget` **(required)**	A local folder or a Git URL.
`apikey` **(required)**	Your Syhunt Web API key.
`sourcebranch`	The branch to scan, for repository URLs.
`sessionname`, `huntmethod`, `reporttemplate`, `reportgl`, `timelimit`	As for dynamic scans.

Response:

```
{
  "huntmethod": "normal",
  "pid": 4476,
  "result": true,
  "resultstr": "",
  "sessionname": "Test2",
  "sessiontype": "code"
}
```

The response keys are the same as for dynamic scans.

Getting results

All result requests go to **`POST /syhunt/results.lua`** and take:

Key	Description
`sessionname` **(required)**	The session whose results you want.
`resulttype` **(required)**	`status`, `report_xml`, `report_json` or `session_log`.
`apikey` **(required)**	Your Syhunt Web API key.

Status

```
{
  "sessionname": "Test",
  "resulttype": "status",
  "apikey": "YOUR_API_KEY"
}
```

Response:

```
{
  "report_xml_available": true,
  "report_json_available": true,
```


Page 99 of 175

Syhunt Hybrid 7.2

```
  "session_log_available": true,
  "sessionname": "Test",
  "status": "Completed"
}
```

Key	Description
`report_xml_available`	`true` if an XML report is available.
`report_json_available`	`true` if a JSON report is available.
`session_log_available`	`true` if a scan log is available.
`status`	The session status: `Scanning`, or `Completed` once the scan has finished.

Report

Request `report_xml` or `report_json`:

```
{
  "sessionname": "Test",
  "resulttype": "report_xml",
  "apikey": "YOUR_API_KEY"
}
```

With `report_xml`, the response body is the scan report in XML:

```
<?xml version="1.0"?>
<report>
  <report_title>Syhunt Scanner Report</report_title>
  <!-- ... -->
  <scanner_version>6.5.0.0</scanner_version>
</report>
```

Log

```
{
  "sessionname": "Test",
  "resulttype": "session_log",
  "apikey": "YOUR_API_KEY"
}
```

The response body is the scan log, in plain text.

Page 100 of 175

Syhunt Hybrid 7.2

Lua API

> **Applies to:** Syhunt Hybrid 6.9.12

Contents

- Introduction
- Launching a dynamic scan
- Launching a source code scan
- Getting results
- Generating a report or export
- Getting session details
- Emailing a report
- Updating preferences
- Other functions

Introduction

Syhunt Hybrid Platinum includes a simple Lua API for launching dynamic and source code scans, retrieving the status, report and log of a scan session, and more. The same operations are available over HTTP through the Web API.

Loading the Syhunt library

Load the Syhunt library before using the API:

```
-- From a GUI app
require "SyMini"
-- From a console app
require "SyMini.Console"
```

Launching a dynamic scan

```
local res = symini.scantask_launch({
  sessionname = "Test",
  starturl = "http://127.0.0.1",
  huntmethod = "appscan",
  reporttemplate = "Complete"
})
if res.result == true then
  print(res.sessionname)
end
```

Key	Description
`starturl` **(required)**	The target URL.
`sessionname`	A unique, alphanumeric session name. If omitted, a random one is generated.
`huntmethod`	A valid hunt method name.
`reporttemplate`	A report template: `Standard`, `Comparison`, `Compliance` or `Complete`.
`reportgl`	If `true`, returns JSON output in GitLab format.
`timelimit`	Maximum scan time, for example `1d`, `3h`, `2h30m`, `50m`. The scan is aborted when it's reached. No limit by default.

Returned table:

Page 101 of 175

Syhunt Hybrid 7.2

Key	Description
`pid`	A unique process ID for the running scan.
`result`	`true` if the scan was launched; `false` if it couldn't be started.
`resultstr`	If `result` is `false`, a description of the error.
`sessionname`	The scan's session name. If you didn't provide one, this is the generated name. Use it in later calls to get the results.
`sessiontype`	The session type: `dynamic` or `code`.

Launching a source code scan

```
-- Scan a local folder
local res = symini.scantask_launch({
  sessionname = "Test",
  sourcetarget = "P:\\Private\\MyWebApp\\",
  huntmethod = "normal",
  reporttemplate = "Complete"
})

-- Scan a repository URL
local res = symini.scantask_launch({
  sessionname = "Test",
  sourcetarget = "https://github.com/drnic/php-helloworld.git",
  sourcebranch = "master",
  huntmethod = "normal",
  reporttemplate = "Complete"
})

if res.result == true then
  print(res.sessionname)
end
```

Key	Description
`sourcetarget` **(required)**	A local folder or a Git URL.
`sourcebranch`	The branch to scan, for repository URLs.
`sessionname`, `huntmethod`, `reporttemplate`, `reportgl`, `timelimit`	As for dynamic scans.

The returned table is the same as for dynamic scans.

Getting results

Status

```
local res = symini.scantask_getresults({
  sessionname = "Test",
  resulttype = "status"
})
```

Key	Description
`sessionname` **(required)**	The session whose results you want.
`resulttype` **(required)**	`status`, `report_xml`, `report_json` or `session_log`.

Returned table:

Page 102 of 175

Key	Description
`report_xml_available`	`true` if an XML report is available.
`report_json_available`	`true` if a JSON report is available.
`session_log_available`	`true` if a scan log is available.
`status`	The session status: `Scanning`, or `Completed` once the scan has finished.

Report

```
-- XML report
local res = symini.scantask_getresults({
  sessionname = "Test",
  resulttype = "report_xml"
})

-- JSON report
local res = symini.scantask_getresults({
  sessionname = "Test",
  resulttype = "report_json"
})
```

Log

```
local res = symini.scantask_getresults({
  sessionname = "Test",
  resulttype = "session_log"
})
```

Returns the scan log as text.

Generating a report or export

```
local res = symini.genreport({
  sessionname = 'mysessionname',
  outfilename = 'myreport.pdf',
  template = 'Standard'
})
```

Key	Description
`sessionname` **(required)**	The session to report on.
`outfilename` **(required)**	The report's output file name; the extension sets the format.
`template`	The report template. Default: `Standard`.
`passfailcond`	A pass/fail condition.

Returned table:

Key	Description
`outfilename`	The report's full output path.
`result`	`true` if the report was generated.
`resultstr`	Status of the report generation.
`passfail_result`	`true` if the pass/fail condition passed.
`passfail_status`	Status of the pass/fail condition.

Syhunt Hybrid 7.2

Key	Description
`passfail_resultstr`	Reason for the pass/fail result.

Getting session details

```
local res = symini.getsessiondetails('mysessionname')
```

Returned table:

Key	Description
`huntmethod`	Hunt method.
`datetime`	Date and time of the scan.
`duration`	Duration of the scan.
`targets`	Target(s) of the scan.
`targetdesc`	Description of the target.
`sourcedir`	Target source code folder, if any.
`targetfile`	Target file, if any.
`targeturl`	Target URL, if any.
`vulncount`	Number of vulnerabilities.
`attackcount`	Number of attacks (web server log scans).
`ports`	Target ports, if any.
`status`	Session status.
`resultsdesc`	Description of the results.
`launcher`	Name of the tool that launched the session.

Emailing a report

```
local res = symini.emailreport({
  tracker = 'myemailtrackername',
  filename = 'myreport.pdf',
  subject = 'My Test Report'
})
```

Key	Description
`tracker` **(required)**	The name of an email tracker you've set up.
`filename` **(required)**	Full path of the report to send.
`subject`	Email subject. Default: `Syhunt Scanner Report`.

Returned table:

Key	Description
`result`	`true` if the email was sent.
`resultstr`	Status, including any error message.

Page 104 of 175

Syhunt Hybrid 7.2

Key	Description
`resultdbg`	Debug log of the operation.

Updating preferences

Function	Description
`symini.prefs_set(key, value [, targeturl])`	Sets a persistent preference, globally or for a target.
`symini.prefs_get(key [, targeturl])`	Gets a persistent preference, globally or for a target.

See Preferences for the available preference IDs.

```
-- Update a site preference
local res = symini.prefs_set('enabled', true, 'http://127.0.0.1')
-- Update a global preference
local res = symini.prefs_set('hybrid.report.company.logo.url', 'https://www.mydomain.com/mylogo.png')
if res.result == true then
  print('Preference updated.')
end
```

Returned table:

Key	Description
`result`	`true` if the operation succeeded.
`resultstr`	Status, including any error message.
`value`	The preference's value (`prefs_get` only).

Other functions

Function	Description
`symini.info.version`	The Syhunt version.
`symini.getsessionname()`	Returns a random session name you can use to launch a scan.
`symini.getsessioncomparison(sessionname1, sessionname2)`	Returns a comparison of two sessions.
`symini.getsessionlist([maxdays])`	Returns the list of past sessions.
`symini.runcmd(cmdname)`	Runs a command (see below).

Commands available to `symini.runcmd` :

Command	Effect
`stop`	Stops all running processes of this Syhunt installation.
`clearinc`	Clears the incremental scan cache.
`clearpref`	Clears the global preferences.
`clearsite`	Clears the site preferences.
`cleartrack`	Removes all issue trackers.

More examples, including object-based usage, are in the scantools repository.

Page 105 of 175

Syhunt Hybrid 7.2

GitHub

> **Applies to:** Syhunt Hybrid 7.2

Contents

- Introduction
- Activating a Syhunt runner
- Adding Syhunt to a workflow
- Scan functions and pass/fail conditions
- Sending vulnerabilities to GitHub Issues

Introduction

You can launch Syhunt scans from a GitHub Actions workflow, bringing Syhunt Dynamic, Syhunt Code and Syhunt Mobile into your continuous delivery pipeline. You can also set up GitHub as an issue tracker in Syhunt, so vulnerabilities can be submitted to a project's Issues.

The following workflow scans the repository's source code, fails the job if medium- or high-risk vulnerabilities are found, and attaches a PDF vulnerability report to the job's artifacts:

```
name: syhunt test
on:
  push:
    branches:
      - main
jobs:
  build:
    runs-on: [self-hosted, Windows, X64]
    steps:
      - uses: actions/checkout@v2
      - run: Start-CodeScan -pfcond 'medium' -output 'report.pdf'
      - name: 'Syhunt Report'
        uses: actions/upload-artifact@v2
        if: failure()
        with:
          name: syhunt-report
          path: report.pdf
          if-no-files-found: error
```

Activating a Syhunt runner

The Syhunt runner is a self-hosted GitHub Actions runner that receives scan requests, runs the scans and reports the results back to GitHub.

> **Important**
> For security and performance, install the runner on a separate virtual machine or a dedicated physical machine.

1. Install Git for Windows with the default settings.
2. Install Syhunt Hybrid (`syhunt-hybrid-<version>.exe`) with the default settings.
3. On GitHub, open the repository's main page and:
    1. Click **Settings**.
    2. In the left sidebar, click **Actions**, then **Runners**.
    3. Click **New self-hosted runner**.

Page 106 of 175

Syhunt Hybrid 7.2

4. Select the **Windows** operating system and the **x64** architecture.

GitHub shows instructions for downloading the runner application and installing it on your machine.

4. Follow those instructions to register the runner with the repository's token. When you run `config.cmd`:
    1. Press **Enter** to add the runner to the Default group.
    2. Enter a name for the runner, for example `syhunt`.
    3. Enter a label for the runner, for example `syhunt`.
    4. The messages **√ Runner successfully added** and **√ Runner connection is good** confirm success.
    5. Press **Enter** to keep `_work` as the work folder.
    6. Choose whether to run the runner as a Windows service.
5. Run `run.cmd`. You should see:

```
√ Connected to GitHub
2022-02-04 19:50:21Z: Listening for Jobs
```

Syhunt can now be called from GitHub Actions workflows.

> **Note**
> GitHub automatically removes a self-hosted runner that hasn't connected to GitHub Actions for more than 30 days. Each workflow run is also limited to 72 hours; runs that reach the limit are cancelled.

Adding Syhunt to a workflow

If your repository doesn't have a workflow file yet, create one, for example `.github/workflows/syhunt.yml`.

SAST example: scans the repository's source code, fails the job if medium- or high-risk vulnerabilities are found, and attaches a PDF report to the job's artifacts:

```yaml
name: syhunt test
on:
  push:
    branches:
      - main
jobs:
  build:
    runs-on: [self-hosted, Windows, X64]
    steps:
      - uses: actions/checkout@v2
      - run: Start-CodeScan -pfcond 'medium' -output 'report.pdf'
      - name: 'Syhunt Report'
        uses: actions/upload-artifact@v2
        if: failure()
        with:
          name: syhunt-report
          path: report.pdf
          if-no-files-found: error
```

DAST example: scans the live web application after deployment, fails the job if medium- or high-risk vulnerabilities are found, and attaches a PDF report to the job's artifacts:

```yaml
name: syhunt test
on:
  push:
    branches:
      - main
jobs:
  build:
    runs-on: [self-hosted, Windows, X64]
    steps:
      - uses: actions/checkout@v2
```


Page 107 of 175

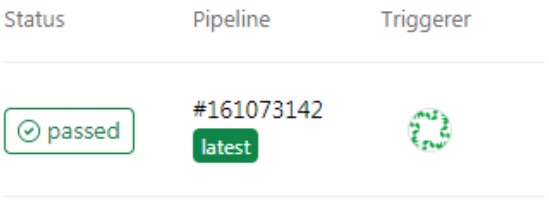
Syhunt Hybrid 7.2

```
      - run: Start-DynamicScan -target 'www.productionurl.com' -pfcond 'medium' -output 'report.pdf'
      - name: 'Syhunt Report'
        uses: actions/upload-artifact@v2
        if: failure()
        with:
          name: syhunt-report
          path: report.pdf
          if-no-files-found: error
```

Scan functions and pass/fail conditions

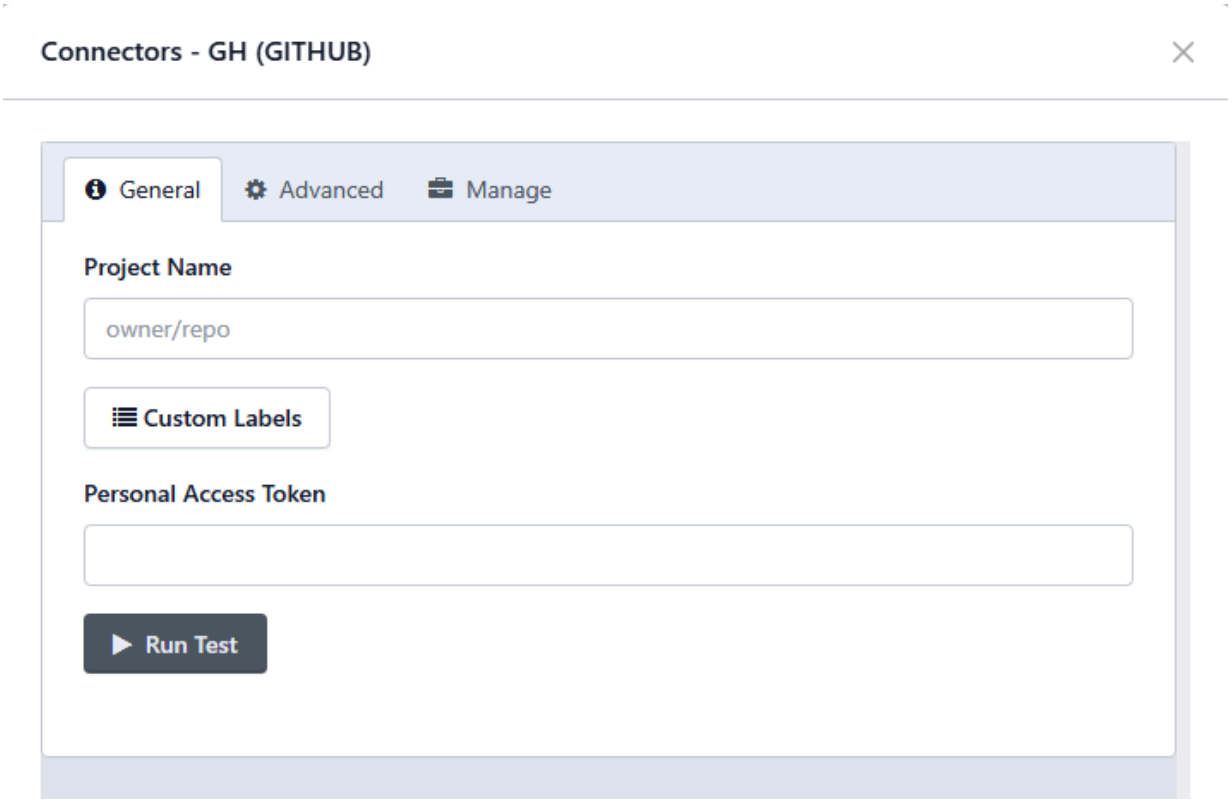
Workflow steps call the Syhunt PowerShell functions. See the PowerShell guide for all parameters and more examples:

- `Start-DynamicScan`
- `Start-CodeScan`
- More examples
- Pass/fail conditions



Sending vulnerabilities to GitHub Issues

Once a GitHub issue tracker is configured, you can submit a vulnerability to a project with one click.



See Issue Trackers: GitHub for how to set up the tracker, and Submitting vulnerabilities to a tracker for how to send findings.

Page 108 of 175

Syhunt Hybrid 7.2

GitLab

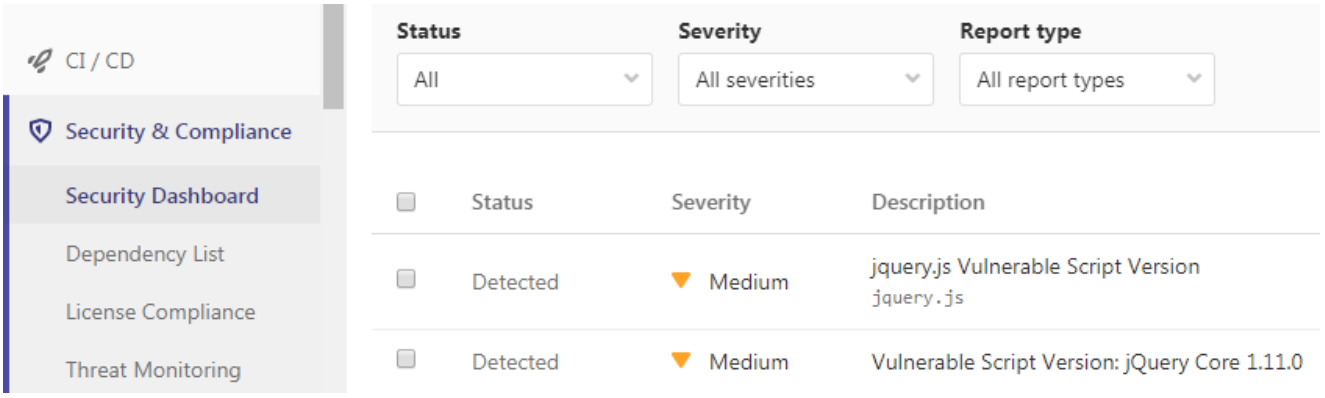
> **Applies to:** Syhunt Hybrid 7.2

Contents

- Introduction
- Activating a Syhunt runner
- Adding Syhunt to your CI file
- GitLab security dashboard
- Scan functions and pass/fail conditions
- Advanced runner settings
- Sending vulnerabilities to GitLab Issues

Introduction

You can launch Syhunt scans from a GitLab CI pipeline, bringing Syhunt Dynamic, Syhunt Code and Syhunt Mobile into your continuous delivery pipeline and security dashboard, scheduling scans and more. You can also set up GitLab as an issue tracker in Syhunt, so vulnerabilities can be submitted to a project's Issues.



The following job scans the repository's source code, fails if medium- or high-risk vulnerabilities are found, attaches a PDF report to the job's artifacts and adds the findings to GitLab's security dashboard:

```
syhunt_test:
  script:
    - Start-CodeScan -pfcond 'medium' -output 'report.pdf' -outputex 'gl-sast-report.json'
  artifacts:
    reports:
      sast: gl-sast-report.json
    paths:
      - report.pdf
    when: on_failure
  tags:
    - syhunt
```

Activating a Syhunt runner

The Syhunt runner is a CI service that receives scan requests, runs the scans and reports the results back to GitLab.

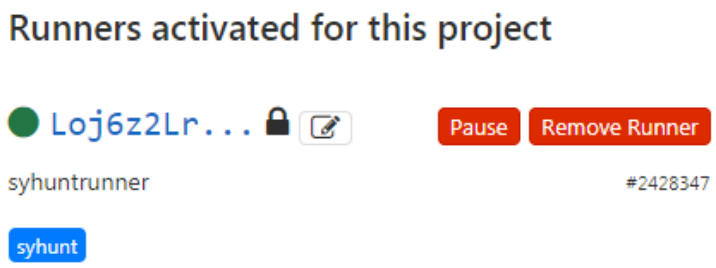
> **Important**
> For security and performance, install the runner on a separate virtual machine or a dedicated physical machine.

Page 109 of 175

Syhunt Hybrid 7.2

1. In your GitLab project, get a runner registration token:
    1. Click **Settings** → **CI/CD**.
    2. Expand **Runners**.
    3. Scroll to **Set up a specific Runner manually**.
    4. Copy the registration token and keep it somewhere safe; you'll need it shortly.
2. Install Git for Windows with the default settings.
3. Install Syhunt Hybrid (`syhunt-hybrid-<version>.exe`) with the default settings.
4. Download and run the Syhunt Runner setup. When prompted for the runner registration details, paste the registration token, click **Next** and complete the installation.

Go back to the **Runners** section in GitLab and refresh the page (F5). `syhuntrunner` should now appear under **Runners activated for this project**:



Syhunt can now be called from your CI pipeline.

Adding Syhunt to your CI file

If your repository doesn't have a CI file yet, go to **Project overview** and click **Set up CI/CD**. This creates a `.gitlab-ci.yml` file in the repository.

SAST example: scans the repository's source code, fails if medium- or high-risk vulnerabilities are found, attaches a PDF report to the job's artifacts and adds the findings to the security dashboard:

```
syhunt_test:
  stage: test
  script:
    - Start-CodeScan -pfcond 'medium' -output 'report.pdf' -outputex 'gl-sast-report.json'
  artifacts:
    reports:
      sast: gl-sast-report.json
    paths:
      - report.pdf
    when: on_failure
  tags:
    - syhunt
```

DAST example: scans the live web application after deployment, fails if medium- or high-risk vulnerabilities are found, attaches a PDF report to the job's artifacts and adds the findings to the security dashboard:

```
production:
  stage: deploy
  script:
    - Start-DynamicScan -target 'www.productionurl.com' -pfcond 'medium' -output 'report.pdf' -outputex 'gl-
dast-report.json'
  artifacts:
    reports:
      dast: gl-dast-report.json
    paths:
      - report.pdf
    when: on_failure
  tags:
    - syhunt
  only:
    - tags
```


Page 110 of 175

Syhunt Hybrid 7.2

For more examples, see the PowerShell guide.

GitLab security dashboard

Syhunt generates a GitLab-compatible vulnerability report when the `-outputex` file name is:

- **SAST:** `gl-sast-report.json` or `<anyname>.gls.json`
- **DAST:** `gl-dast-report.json` or `<anyname>.gld.json`

To enable the integration, also add the file to `artifacts.reports.sast` or `artifacts.reports.dast` in your CI file, as in the examples above.

How GitLab displays the results

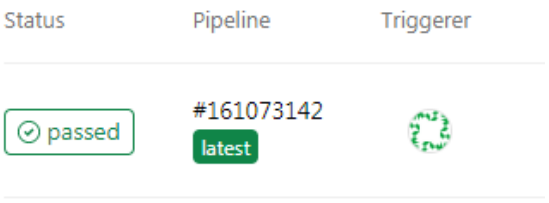
- GitLab currently adds only **successful** pipelines to a project's security dashboard and vulnerability report. To see every finding there, omit `-pfcond` from the Syhunt step. Alternatively, with a `high` pass/fail condition, if no high-risk vulnerabilities are found, all medium- to info-level findings appear in the dashboard.
- The pipeline's **Security** tab always lists every vulnerability, whether the pipeline succeeded or failed. However, if the pipeline failed because a pass/fail condition was met, GitLab may inconsistently show a count of 0 vulnerabilities while still listing them. This is a known GitLab issue.

> **Tip**
> If this is the first scan of a **large** codebase, scan it once without the dashboard integration to check how many vulnerabilities there are. If there are many, fix the first batch reported by Syhunt to improve the application's security, and only then enable the dashboard integration.

Scan functions and pass/fail conditions

CI jobs call the Syhunt PowerShell functions. See the PowerShell guide for all parameters:

- `Start-DynamicScan`
- `Start-CodeScan`
- Pass/fail conditions



Advanced runner settings

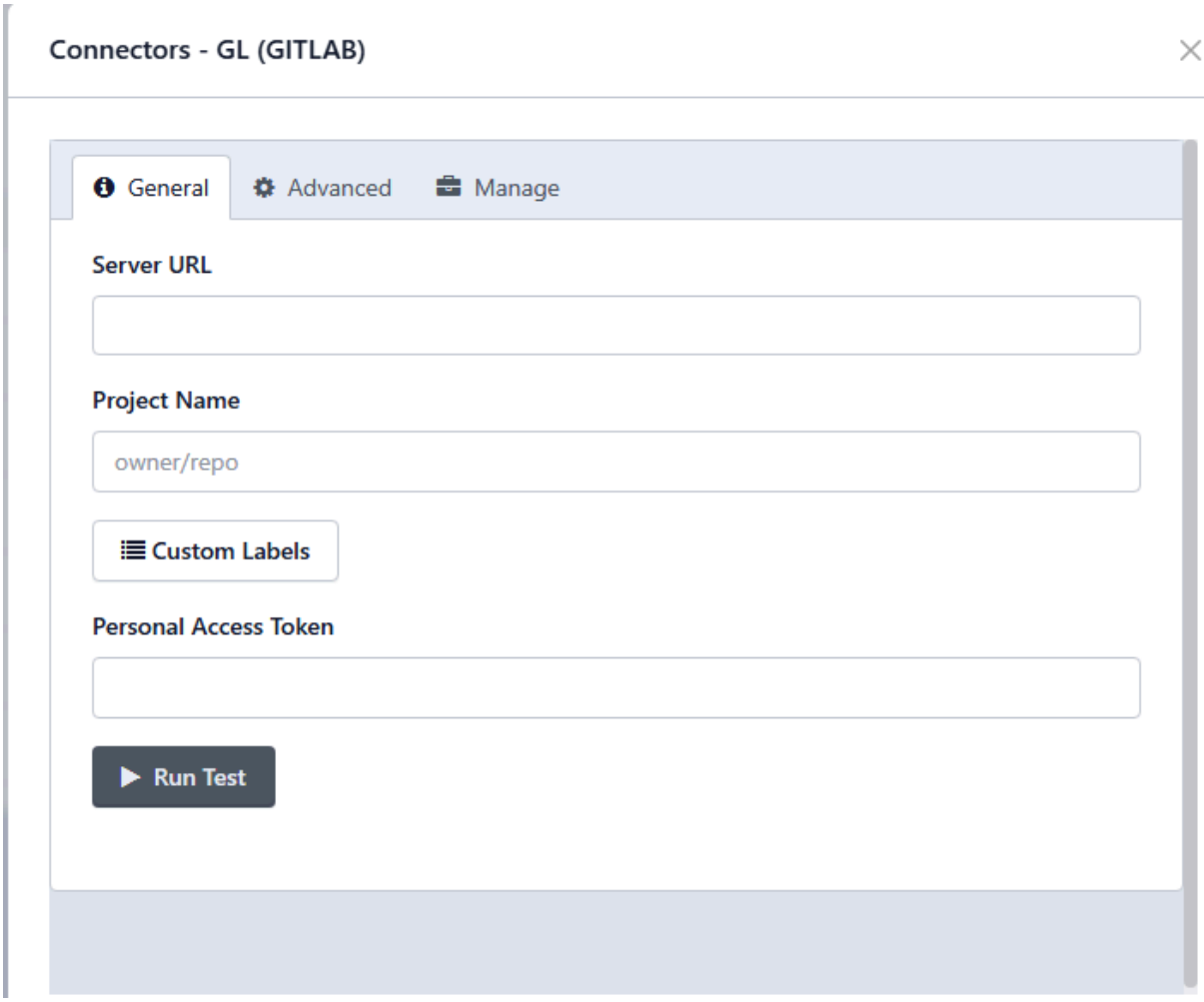
To let the runner process several jobs at once, increase the `concurrent` value in `C:\SyhuntRunner\config.toml`. See GitLab's advanced configuration documentation.

Sending vulnerabilities to GitLab Issues

Once a GitLab issue tracker is configured, you can submit a vulnerability to a project with one click.

Page 111 of 175

Syhunt Hybrid 7.2



See Issue Trackers: GitLab for how to set up the tracker, and Submitting vulnerabilities to a tracker for how to send findings.

Page 112 of 175

Syhunt Hybrid 7.2

Bitbucket

> **Applies to:** Syhunt Hybrid 7.2

Contents

- Introduction
- Activating a Syhunt runner
- Adding Syhunt to your pipeline
- Scan functions and pass/fail conditions
- Bitbucket Issues

Introduction

You can launch Syhunt scans from a Bitbucket Pipelines file, bringing Syhunt Dynamic, Syhunt Code and Syhunt Mobile into your continuous delivery pipeline. Runners are supported on both Windows and Linux.

The following pipelines scan the repository's source code, fail the step if medium- or high-risk vulnerabilities are found, and attach a vulnerability report to the step's artifacts.

Windows runner:

```
pipelines:
  default:
    - step:
        runs-on:
          - self.hosted
          - windows
        script:
          - Start-CodeScan -pfcond 'fail-if:risk=mediumup' -output 'report.pdf'
        artifacts:
          - report.pdf
```

Linux runner:

```
pipelines:
  default:
    - step:
        runs-on:
          - self.hosted
          - linux.shell
        script:
          - ${HOME}/syhunt-hybrid/carbon/scancode ${PWD} -pfcond:fail-if:risk=mediumup -
rout:${PWD}/report.html
        artifacts:
          - report.html
```

Activating a Syhunt runner

The Syhunt runner is a self-hosted Bitbucket runner that receives scan requests, runs the scans and reports the results back to Bitbucket.

> **Important**
> For security and performance, install the runner on a separate virtual machine or a dedicated physical machine.

1. Install Git with the default settings:

Page 113 of 175

- **Windows:** Git for Windows.
- **Linux:** the `git` package from your distribution's package manager.

2. Install Syhunt Hybrid with the default settings (`syhunt-hybrid-<version>.exe` on Windows, `syhunt-hybrid-<version>.jar` on Linux).

3. Run `java -version` in a Command Prompt or terminal and check that Java 22.0.2 or later is installed. If not, install JDK 22 or later.

4. On Bitbucket, open the repository and:

    1. Click **Repository settings** in the left sidebar.
    2. Scroll down to **Pipelines** and click **Runners**.
    3. Click **Add runner**. Under **System and architecture**, select **Windows (64-bit)**, or **Linux Shell** if Syhunt is installed on Linux.
    4. Give the runner a name (for example, `Syhunt`) and a label (for example, `syhunt`), then click **Next**.

    Bitbucket shows instructions for downloading the runner application and installing it on your machine.

5. Follow those instructions to activate the runner, then click **Next**.

Syhunt can now be called from your Bitbucket pipelines.

Adding Syhunt to your pipeline

If your repository doesn't have a pipeline file yet, create `bitbucket-pipelines.yml` in its root.

SAST

Scans the repository's source code, fails the step if medium- or high-risk vulnerabilities are found and attaches a report to the step's artifacts.

Windows:

```
pipelines:
  default:
    - step:
        runs-on:
          - self.hosted
          - windows
        script:
          - Start-CodeScan -pfcond 'fail-if:risk=mediumup' -output 'report.pdf'
        artifacts:
          - report.pdf
```

Linux:

```
pipelines:
  default:
    - step:
        runs-on:
          - self.hosted
          - linux.shell
        script:
          - ${HOME}/syhunt-hybrid/carbon/scancode ${PWD} -pfcond:fail-if:risk=mediumup -
rout:${PWD}/report.html
        artifacts:
          - report.html
```

DAST

Scans the live web application after deployment, fails the step if medium- or high-risk vulnerabilities are found and attaches a report to the step's artifacts.

Syhunt Hybrid 7.2

Windows:

```
pipelines:
  default:
    - step:
        runs-on:
          - self.hosted
          - windows
        script:
          - Start-DynamicScan -target 'www.productionurl.com' -pfcond 'medium' -output 'report.pdf'
        artifacts:
          - report.pdf
```

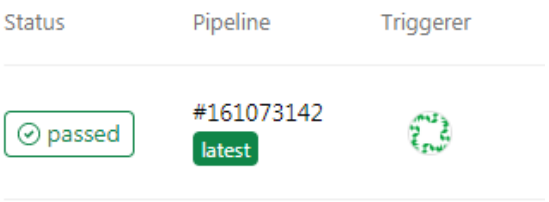
Linux:

```
pipelines:
  default:
    - step:
        runs-on:
          - self.hosted
          - linux.shell
        script:
          - ${HOME}/syhunt-hybrid/carbon/scanurl www.productionurl.com -pfcond:fail-if:risk=mediumup -
rout:${PWD}/report.html
        artifacts:
          - report.html
```

Scan functions and pass/fail conditions

	Windows runner	Linux runner
Dynamic scans	`Start-DynamicScan` (PowerShell)	`scanurl`
Code scans	`Start-CodeScan` (PowerShell)	`scancode`
Pass/fail conditions	PowerShell	CLI

More Windows examples are in the PowerShell guide.



Bitbucket Issues

Integration with Bitbucket Pipelines is available now. Integration with Bitbucket Issues is planned for a future release.

Page 115 of 175

Syhunt Hybrid 7.2

Azure DevOps

> **Applies to:** Syhunt Hybrid 7.2

Since version 6.9.6, Syhunt Community and Syhunt Hybrid can connect to Azure DevOps Services, Azure DevOps Server and Team Foundation Server (TFS) and scan the source code of projects for application security vulnerabilities and weaknesses.

This feature is available from the user interface and the CLI on Windows, and works with on-premises and cloud-hosted projects, both TFVC and Git. Out of the box, Syhunt supports Azure DevOps Services, Azure DevOps Server 2020 and 2019, and TFS 2018 down to 2015. As explained below, it can also be configured to connect to older TFS versions (2013, 2012 and 2010).

For pipeline automation, see also the PowerShell integration.

> **Note**
> TFS support in Syhunt Community and Hybrid relies on the export tool developed by Microsoft MVP Neno Loje, freely available from his website. It is bundled with Syhunt with the developer's permission.

Supported versions

Version	Deployment
Azure DevOps Services 2021	Cloud
Azure DevOps Server 2020 / 2019	On-premises
Team Foundation Server 2018 / 2015	On-premises
Team Foundation Server 2013 / 2012 / 2010	On-premises*

* Scanning projects on TFS 2010–2013 requires Microsoft Visual Studio or Microsoft Team Explorer on the same machine as Syhunt.

To scan Git repositories (not only TFVC), install Git for Windows on the same machine.

Project URL formats

Repository	Example target
Azure DevOps Services, TFVC	`https://dev.azure.com/user/projectname`
Azure DevOps Services, Git	`https://dev.azure.com/user/projectname/_git/projectname`
Azure DevOps Server	`https://myserver/tfs/projectname`
Azure DevOps Services / Server / TFS, by collection	`collection:https://dev.azure.com/user$/projectname`

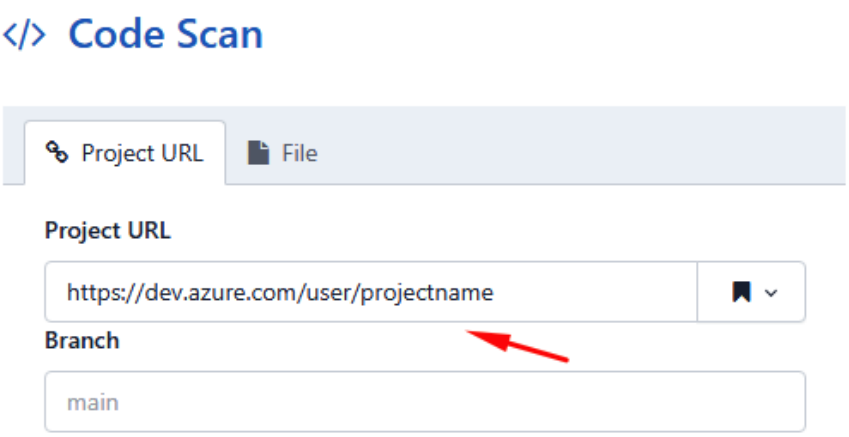
With the `collection:` form, Syhunt connects to the project collection at `https://dev.azure.com/user` and scans the server path `$/projectname`.

Scanning from the user interface

In the Syhunt Hybrid Web UI (or the Community GUI), click **New Scan**, select **Project URL** and enter the project's URL, as described in Source Code Scanning. This works for Azure DevOps and TFS/TFVC projects as well as Git repositories hosted in Azure DevOps. You can also schedule code scans of project URLs with the scan scheduler.

Page 116 of 175

Syhunt Hybrid 7.2



Syhunt asks for your Azure DevOps credentials before connecting, unless you've already authenticated.

To connect to an older TFS server (2010 to 2013), choose its version on the **Advanced** tab. As noted above, this requires Visual Studio or Team Explorer on the same machine.

Scanning from the command line

```
scancode https://dev.azure.com/user/projectname
scancode https://myserver/tfs/projectname
scancode collection:https://dev.azure.com/user$/projectname
```

Syhunt asks for your Azure DevOps credentials before connecting, unless you've already authenticated.

For older TFS servers, specify the version with `-tfsv`:

```
scancode https://myserver/tfs/projectname -tfsv:2013
scancode https://myserver/tfs/projectname -tfsv:2012
scancode https://myserver/tfs/projectname -tfsv:2010
```

See the CLI guide for more examples.

Pipeline integration

You can automate Azure DevOps scans with:

- **PowerShell**: see the PowerShell guide.
- **The CLI**: as shown above and in the CLI guide.

Page 117 of 175

Syhunt Hybrid 7.2

Jenkins

> **Applies to:** Syhunt Hybrid 7.2

Contents

- Introduction
- Activating a Syhunt agent
- Before installing Syhunt (Windows)
- Before installing Syhunt (Linux)
- Adding Syhunt to a pipeline
- `scanURL` function
- `scanCode` function
- Pass/fail conditions
- Scheduling scans

Introduction

You can run Syhunt scans from a Jenkins pipeline script, bringing Syhunt Dynamic, Syhunt Code and Syhunt Mobile into your continuous delivery pipeline, scheduling scans and more.

Where to install Syhunt

Syhunt can be installed on either:

1. **The Jenkins machine itself**, so that Syhunt and the Jenkins server run side by side; or
2. **A separate machine acting as a Jenkins agent**, which the main Jenkins server calls remotely over SSH.

Using CI agents is common best practice in DevOps environments, so we strongly recommend the second option. For security and performance, production environments should also use a dedicated machine as the agent.

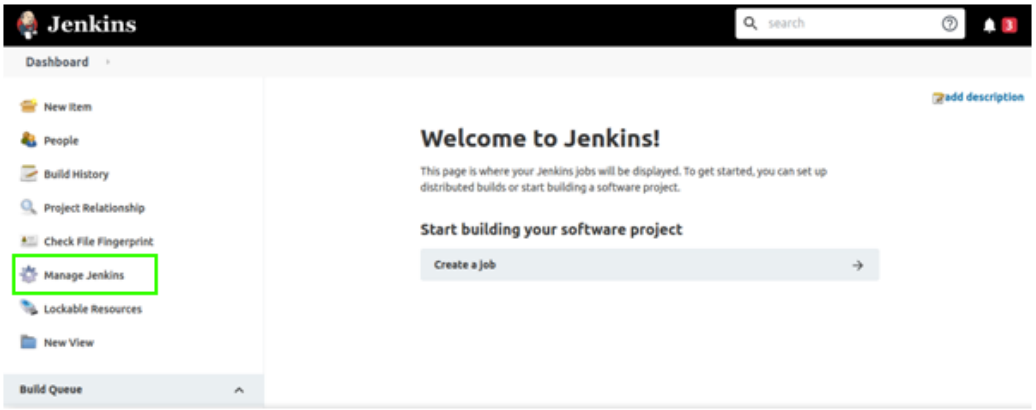
On Windows or Linux, **install Syhunt after Jenkins is deployed and set up**. The Syhunt installer then detects Jenkins and installs the required extensions automatically. If you installed Syhunt first, just run the Syhunt installer again.

Remember that Syhunt is licensed per device: each agent machine needs its own license.

Activating a Syhunt agent

Once both the main (controller) and agent Jenkins installations are deployed and set up, make the agent available to the controller:

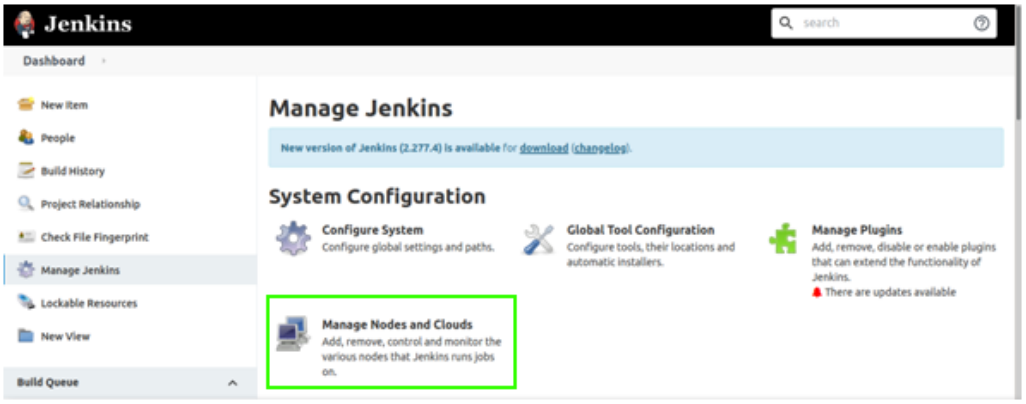
1. On the controller's Jenkins dashboard, click **Manage Jenkins** in the sidebar.



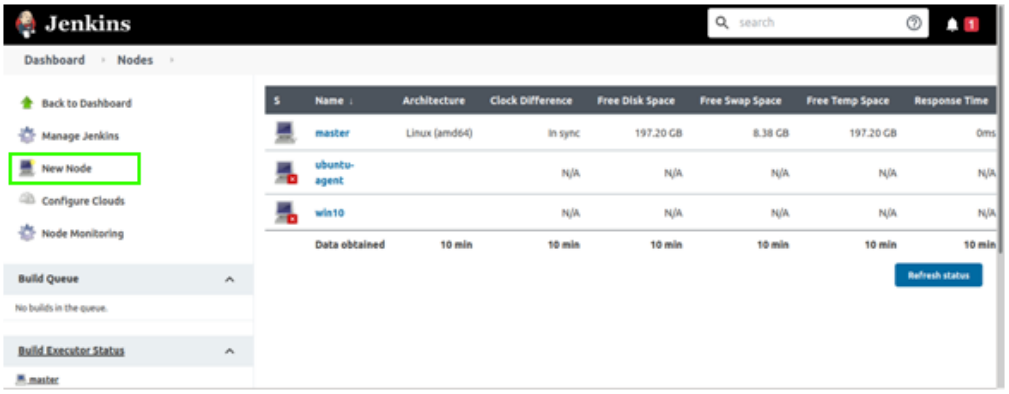
Page 118 of 175

Syhunt Hybrid 7.2

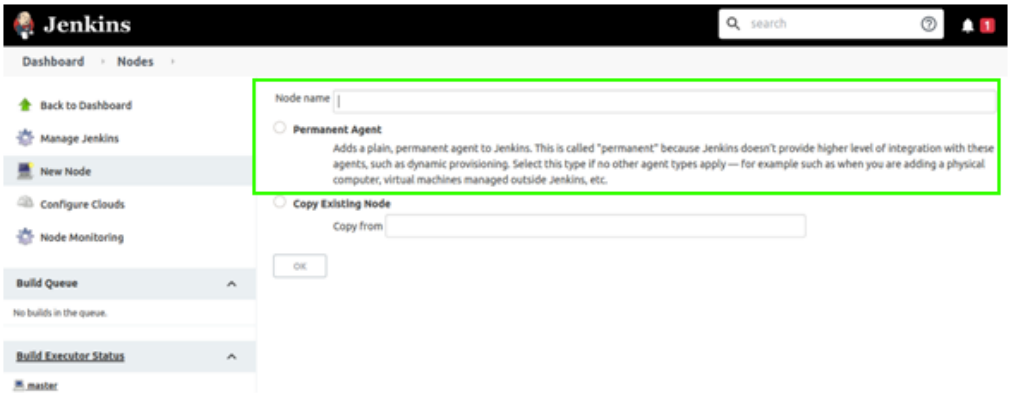
2. Under **System Configuration**, click **Manage Nodes and Clouds**.



3. On the **Nodes** screen, click **New Node** in the sidebar.



1. Name the node.



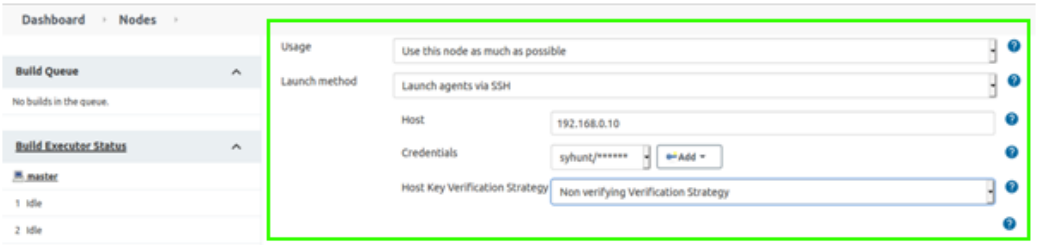
2. Select **Permanent Agent** and click **OK**.
3. Set **# of executors** to **2**.

4. Set **Remote root directory** to the Jenkins directory on the agent: `/var/lib/jenkins` on Linux. On Windows, use the `.jenkins` directory, usually `C:\Users\<youruser>\.jenkins` or `C:\windows\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins`; check which one exists on the agent.
5. Give the node a label you'll use to refer to it later, for example `Syhunt`.

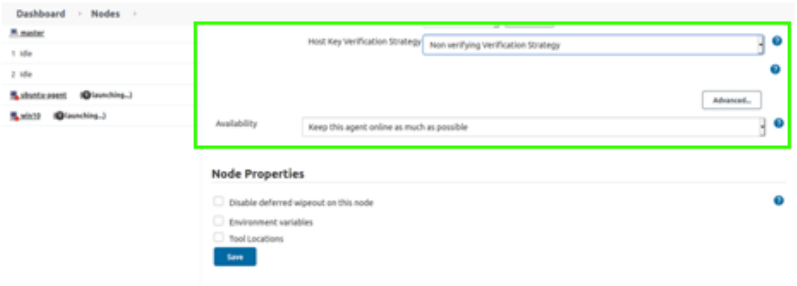
Page 119 of 175

Syhunt Hybrid 7.2

6. Leave **Usage** at its default.



7. Set **Launch method** to **Launch agents via SSH**.
8. Enter the agent machine's host name or IP address.
9. Under **Credentials**, click **Add** and enter the username and password of the agent's Jenkins user.
10. Set the **Host Key Verification Strategy**:
    - If you don't have an SSH credential, use **Non verifying Verification Strategy**.



    - If you created a Jenkins SSH credential on the agent, as described in the Jenkins documentation, use **Manually provided key Verification Strategy**. Using a verification strategy is recommended for better security, especially in production.
11. Leave **Availability** at its default.
12. Click **Save**.

4. If you haven't already, install Syhunt on the agent machine as described below.

Before installing Syhunt (Windows)

After deploying and setting up Jenkins:

1. If you'll use `scanCode()` to scan Git URLs, install Git for Windows.
2. Install Syhunt as described in the installation guide.

Before installing Syhunt (Linux)

After deploying and setting up Jenkins, change the Jenkins user to your current user (as long as it isn't `root`) or another user of your choice. We advise against running Jenkins as `root`, which could expose the operating system and the Jenkins repositories.

1. Edit the Jenkins configuration file:
    - CentOS: `sudo vi /etc/sysconfig/jenkins`
    - Ubuntu: `sudo vi /etc/default/jenkins`
2. Change `JENKINS_USER="jenkins"` to `JENKINS_USER="userofyourchoice"`.
3. Give the new user ownership of the main Jenkins directories:

```
sudo chown -R userofyourchoice:userofyourchoice /var/lib/jenkins
sudo chown -R userofyourchoice:userofyourchoice /var/log/jenkins
sudo chown -R userofyourchoice:userofyourchoice /var/cache/jenkins
```

4. Restart Jenkins:

```
sudo systemctl restart jenkins
```


Page 120 of 175

5. Logged in as that user, install Syhunt as described in the Linux installation guide. If you'll use `scanCode()` to scan Git URLs, install Git as well (also covered there).

Adding Syhunt to a pipeline

1. On the Jenkins dashboard, open the pipeline.
2. Click **Configure** in the sidebar.
3. On the **Pipeline** tab, clear **Use Groovy Sandbox** if it's selected.
4. Add code like this at the appropriate point of your pipeline script:

```
def rootDir = pwd()
def syhunt = load "${rootDir}/../workspace@script/syhunt/syhunt.groovy"
syhunt.scanURL([target: 'http://somewebsite.com/', huntMethod: 'appscan', pfcond: 'fail-
if:risk=mediumup'])
```

   A complete example:

```
#!/usr/bin/env groovy

// If you're not using an agent, change the next line to just: node {
node ("Syhunt") {
    stage('Scan') {
        def rootDir = pwd()
        def syhunt = load "${rootDir}/../workspace@script/syhunt/syhunt.groovy"
        syhunt.scanURL([target: 'http://somewebsite.com/', huntMethod: 'appscan', pfcond: 'fail-
if:risk=mediumup'])
    }
}
```

5. Click **Save**.

Running the build

When the pipeline runs, its console output will look something like this:

```
[Pipeline] echo
Preparing to scan URL: http://somewebsite.com/
[Pipeline] echo
VULNERABLE!
Found 2 vulnerabilities
SYHUNT URLSCAN 6.8.5.4 PLATINUM EDITION (c) 2020 Syhunt
...
[Pipeline] echo
Build problem: found Medium risk vulnerabilities.
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: FAILURE
```

`scanURL` function

Launches a Syhunt Dynamic scan.

Parameter	Description
`target` **(required)**	The URL to scan, for example `http://www.somesite.com`.
`huntMethod`	The hunt method to use. If omitted, the default method is used.

Syhunt Hybrid 7.2

Parameter	Description
`pfcond`	Fails the build when a pass/fail condition is met.
`timelimit`	Maximum scan time, for example `1d`, `3h`, `2h30m`, `50m`. The scan is aborted when it's reached. No limit by default.

Returns a map with:

Key	Description
`outFilename`	The generated report's file name.
`outResultHTML`	The generated report's contents, as HTML text.

`scanCode` function

Launches a Syhunt Code scan.

Parameter	Description
`target` **(required)**	A repository URL, or a local source code folder or file.
`huntMethod`	The hunt method to use. If omitted, the default method is used.
`pfcond`	Fails the build when a pass/fail condition is met.
`branch`	The Git branch to scan. If omitted, the Git client fetches the default branch. Not needed for local folders or files.
`timelimit`	Maximum scan time, for example `1d`, `3h`, `2h30m`, `50m`. No limit by default.

Returns the same map as `scanURL()`.

```
syhunt.scanCode([target: 'https://github.com/someuser/somerepo.git', huntMethod: 'normal', pfcond: 'fail-if:risk=mediumup'])
syhunt.scanCode([target: 'C:\\www\\', huntMethod: 'normal', pfcond: 'fail-if:risk=mediumup'])
```

Pass/fail conditions

Condition	Fails if...
`high` or `fail-if:risk=high`	a High-risk vulnerability or threat is found.
`medium` or `fail-if:risk=mediumup`	a Medium- or High-risk vulnerability or threat is found.
`low` or `fail-if:risk=lowup`	a Low-, Medium- or High-risk vulnerability or threat is found.

Scheduling scans

1. On the Jenkins dashboard, open the pipeline.
2. Click **Configure** in the sidebar.
3. On the **Build Triggers** tab, select **Build periodically** and enter the schedule.
4. Click **Save**.

Page 122 of 175

Syhunt Hybrid 7.2

Issue Trackers

> **Applies to:** Syhunt Hybrid 7.2

Contents

- GitHub Issues
- GitLab Issues
- Jira
- Submitting vulnerabilities to a tracker
- Short names for tracker preferences

Once an issue tracker is configured in Syhunt, you can submit vulnerabilities to a project with one click. Trackers are created as *connectors* in the Web UI; you can also manage them from the command line.

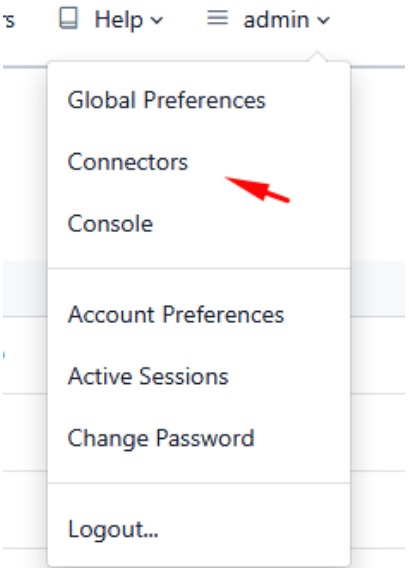
GitHub Issues

1. Create a personal access token with the `repo` scope, if you don't have one:

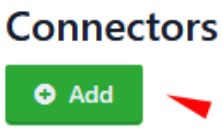
1. Log in to GitHub.
2. Click your avatar (top-right corner) and choose **Settings**.
3. In the left menu, scroll down and choose **Developer settings**.
4. Choose **Personal access tokens**, then **Generate new token**.
5. Optionally, enter a note for the token.
6. Select the `repo` scope.
7. Click **Generate token**.
8. Store the token somewhere safe. Once you leave or refresh the page, you won't be able to see it again.

2. Add a GitHub tracker:

1. Open **Connectors** in the Web UI.



2. Click **Add**.

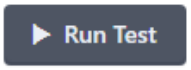


3. Enter a name for the tracker (for example, `MyProjectName`).

Page 123 of 175

Syhunt Hybrid 7.2

4. Set the type to **GitHub**.
5. Click **Create**, then **Close**.
6. Click the new tracker in the list. Its preferences screen opens.
7. Enter the GitHub project in the format `owner/repo`.
8. Enter your personal access token and click **Save**.
9. Open the tracker again and click **Run Test**.



If everything is set up correctly, a test issue is created at `https://github.com/<owner>/<repo>/issues`. Otherwise, an error message suggests what to fix.

GitLab Issues

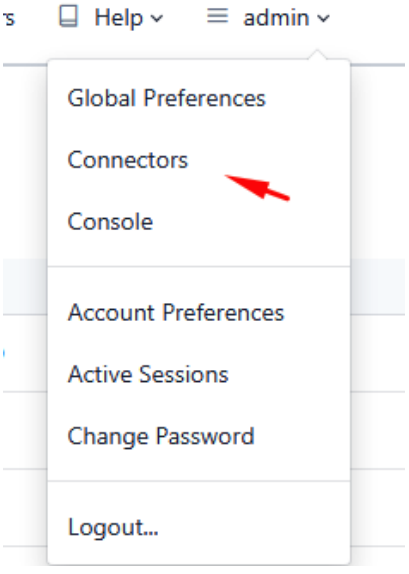
> **Tip**
> To integrate Syhunt with GitLab CI and the GitLab security dashboard, see GitLab.

1. Create a personal access token with the `api` scope, if you don't have one:

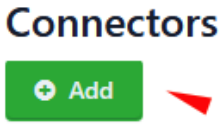
1. Log in to GitLab.
2. Click your avatar (top-right corner) and choose **Settings**.
3. In the User Settings menu, choose **Access Tokens**.
4. Enter a name and, optionally, an expiry date.
5. Select the `api` scope.
6. Click **Create personal access token**.
7. Store the token somewhere safe. Once you leave or refresh the page, you won't be able to see it again.

2. Add a GitLab tracker:

1. Open **Connectors** in the Web UI.



2. Click **Add**.

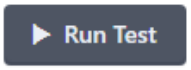


3. Enter a name for the tracker (for example, `MyProjectName`).
4. Set the type to **GitLab**.

Page 124 of 175

Syhunt Hybrid 7.2

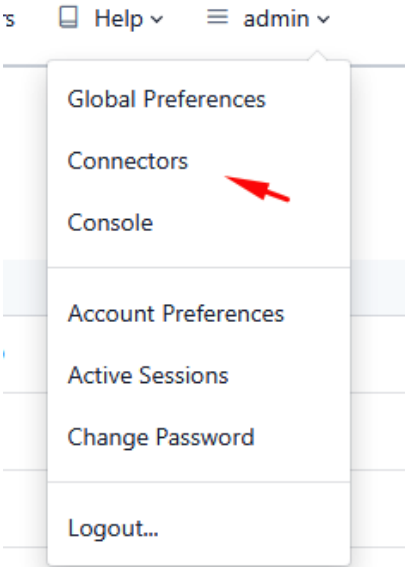
5. Click **Create**, then **Close**.
6. Click the new tracker in the list. Its preferences screen opens.
7. Enter the GitLab project in the format `owner/repo`.
8. Enter the GitLab server URL, for example `https://gitlab.com/` or your own server's URL.
9. Enter your personal access token and click **Save**.
10. Open the tracker again and click **Run Test**.



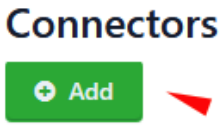
If everything is set up correctly, a test issue is created at `https://<gitlab_server>/<owner>/<repo>/issues`. Otherwise, an error message suggests what to fix.

Jira

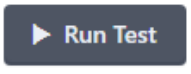
1. Open **Connectors** in the Web UI.



2. Click **Add**.



3. Enter a name for the tracker (for example, `MyProjectName`).
4. Set the type to **JIRA**.
5. Click **Create**, then **Close**.
6. Click the new tracker in the list. Its preferences screen opens.
7. Enter a valid Jira project name.
8. Enter the Jira server URL, for example `http://<IP>:8080/`.
9. If you don't want the default issue type, enter a valid issue type name.
10. Enter your Jira credentials (username and password) and click **Save**.
11. Open the tracker again and click **Run Test**.



If everything is set up correctly, a test issue is created in your Jira project. Otherwise, an error message suggests what to fix.

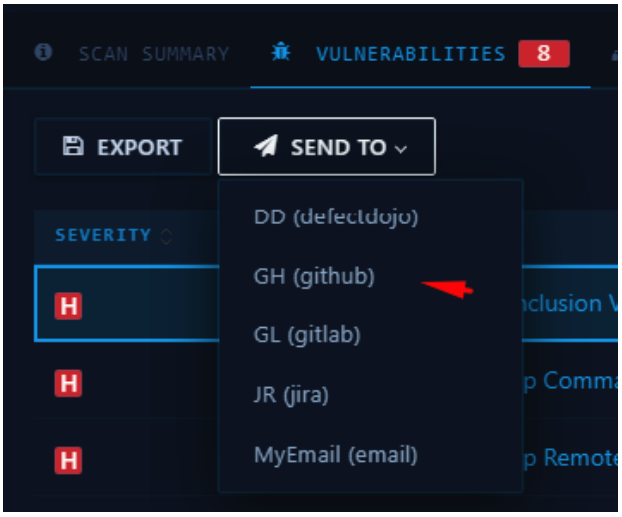
Submitting vulnerabilities to a tracker

1. Open the **Vulnerabilities** tab of a running scan, or open **Scans**, click a scan in the list and go to its **Vulnerabilities** tab.

Page 125 of 175

Syhunt Hybrid 7.2

2. Select one or more vulnerabilities, click **Send To** and choose the tracker.



Syhunt can also send a summary of the vulnerabilities found to a tracker automatically:

- from a **scheduled scan**, if configured on its schedule screen;
- from the **CLI**, with the `-tk` parameter of `scanurl` or `scancode` (see Command Line);
- from **PowerShell** or a **CI workflow**, with the `-tracker` parameter:

```
# PowerShell, existing tracker
Start-CodeScan -pfcond "medium" -tracker "YOURTRACKERNAME"

# PowerShell, tracker defined on the fly
Start-CodeScan -pfcond "medium" -tracker "?app=github###project=user/project###token=YOURTOKEN"

# GitHub Actions workflow, tracker defined on the fly
Start-CodeScan -pfcond "medium" -tracker "?app=github###project=${{ github.repository }}###token=${{
secrets.GITHUB_TOKEN }}"
```

Short names for tracker preferences

When you pass tracker preferences through `-tk` or `-tracker`, use the short names below. Settings without a short name can't be set through these parameters.

Short name	Preference ID	Type
`app`	`tracker.defaultapp`	string
`project`	`tracker.project.name`	string
`token`	`tracker.auth.token.encrypted`	string
`from`	`tracker.message.from`	string
`to`	`tracker.message.tolist`	string
`subject`	`tracker.message.subject`	string
`notifyonfailonly`	`tracker.notifyonfailonly`	boolean
`apiurl`	`tracker.api.url`	string

DefectDojo-specific:

Short name	Preference ID	Type
`producttypename`	`tracker.scan.producttypename`	string

Page 126 of 175

Syhunt Hybrid 7.2

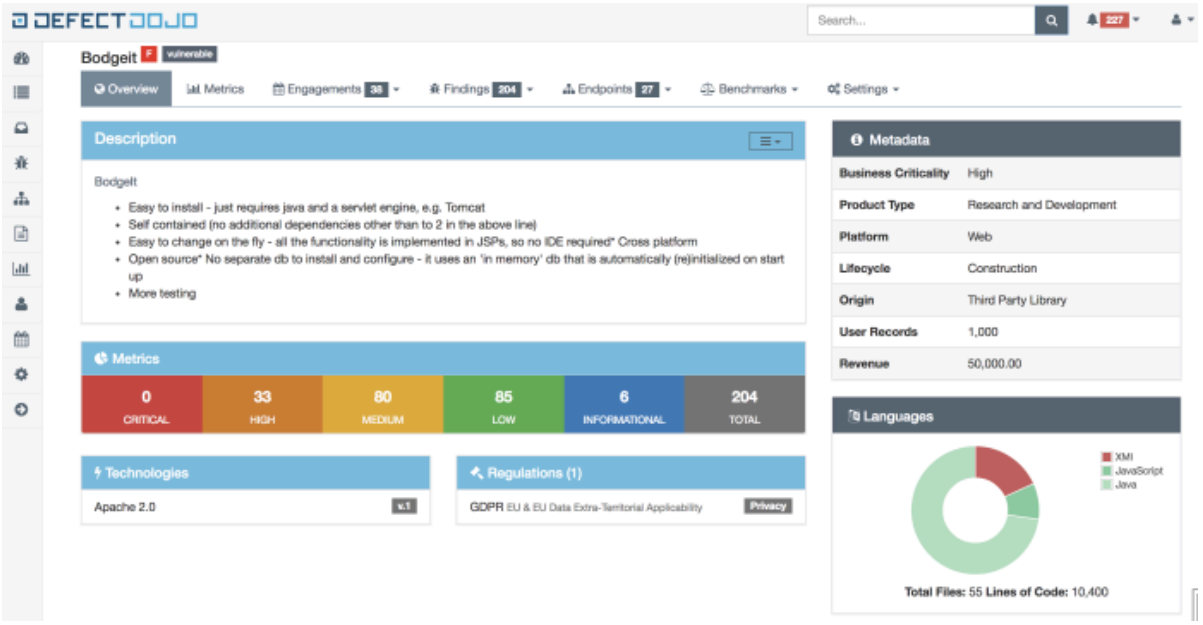
Short name	Preference ID	Type
`testtitle`	`tracker.scan.testtitle`	string
`engagementname`	`tracker.scan.engagementname`	string
`productname`	`tracker.scan.productname`	string
`minseverity`	`tracker.scan.minseverity`	string
`pushtojira`	`tracker.scan.pushtojira`	boolean
`verified`	`tracker.scan.verified`	boolean
`closeoldfindings`	`tracker.scan.closeoldfindings`	boolean
`skipduplicates`	`tracker.scan.skipduplicates`	boolean
`active`	`tracker.scan.active`	boolean
`autocreatecontext`	`tracker.scan.autocreatecontext`	boolean

Page 127 of 175

Syhunt Hybrid 7.2

DefectDojo

> **Applies to:** Syhunt Hybrid 7.2



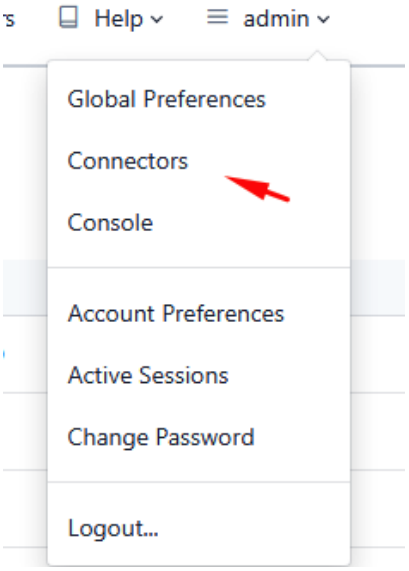
DefectDojo automates application security vulnerability management. It streamlines security testing with features such as importing third-party findings, merging and de-duplication, Jira integration, templating, report generation and security metrics.

Since version 6.9.17, Syhunt can connect to a DefectDojo server's API and submit SAST or DAST results automatically. Alternatively, Syhunt can generate a compatible report that you import into DefectDojo manually.

Submitting results automatically

1. Create a DefectDojo connector

1. Click **Connectors** in the Web UI's top menu.



2. Click **Add**. The **New Connector** screen opens.
3. Enter a name for the connector (for example, `MyDefectDojo`).
4. Set **Type** to **DefectDojo**.

Page 128 of 175

Syhunt Hybrid 7.2

5. Click **Create** and, after the success message, **Close**.

2. Configure it

1. Click the connector in the list. Its preferences screen opens.
2. Enter the DefectDojo server URL, for example `https://<IP>/`.
3. Enter your API token.
4. Click **Save**.

3. Test it

Click the connector in the list and choose **Run Test**. If everything is set up correctly, a test scan appears at `https://<server>/engagement/active`. Otherwise, an error message suggests what to fix.

4. Attach it to your scans

- **CLI:** pass `-tk:<connectorname>`.
- **Scheduled scans:** in the scheduled scan's preferences, on the **Notifications** tab, select the connector under **Issue Tracker #1** or **#2**.

Creating the connector from the CLI

```
-- Add a DefectDojo tracker
scancore -tracker:add
-- Enter the type (DefectDojo) and a name for the tracker, then press Enter.

-- Configure it
scancore -tracker:set to:mytrackername -key:api.url -v:"https://serverurl/"
scancore -tracker:set to:mytrackername -key:auth.token.encrypted -vsecret

-- Test it
scancore -tracker:send -tid:TEST -to:mytrackername

-- Optional settings
scancore -tracker:set to:mytrackername -key:scan.minseverity -v:Info
scancore -tracker:set to:mytrackername -key:scan.testtitle -v:"Syhunt Scan"
scancore -tracker:set to:mytrackername -key:scan.engagementname -v:"Syhunt Hybrid Scan"
scancore -tracker:set to:mytrackername -key:scan.producttypename -v:"Research and Development"
scancore -tracker:set to:mytrackername -key:scan.productname -v:Syhunt
scancore -tracker:set to:mytrackername -key:scan.autocreatecontext -v:true
scancore -tracker:set to:mytrackername -key:scan.active -v:true
scancore -tracker:set to:mytrackername -key:scan.verified -v:true
scancore -tracker:set to:mytrackername -key:scan.closeoldfindings -v:false
scancore -tracker:set to:mytrackername -key:scan.skipduplicates -v:false
scancore -tracker:set to:mytrackername -key:scan.pushtojira -v:false
```

Then pass the tracker to PowerShell or the CLI:

```
# PowerShell: existing tracker with its saved settings
Start-CodeScan -pfcond "medium" -tracker "mytrackername"

# PowerShell: existing tracker, overriding settings for this run only
Start-CodeScan -pfcond "medium" -tracker "mytrackername?productname=Syhunt###autocreatecontext=true"

# PowerShell: tracker defined on the fly
Start-CodeScan -pfcond "medium" -tracker "?
app=defectdojo###apiurl=https://serverurl/###token=YOURTOKEN###productname=Syhunt###testtitle=Syhunt
Scan###autocreatecontext=true"
```

From the CLI, use `-tk` instead of `-tracker`:

```
scancode <target> -pfcond:medium -tk:"mydefectdojotrackername?productname=Syhunt###autocreatecontext=true"
```


Page 129 of 175

Syhunt Hybrid 7.2

> **Note**
> In `-tk` and `-tracker`, preferences must use their short names.

Generating a compatible report

To import results manually, have Syhunt generate a DefectDojo-compatible report by giving the export file name the double extension `.sast.json` or `.dast.json`. When importing the file into DefectDojo, choose the **GitLab SAST** or **GitLab DAST** format.

Scan	Example
SAST	`scancode <target> -xout:anyfilename.sast.json`
DAST	`scanurl <target> -xout:anyfilename.dast.json`

- **GitHub, GitLab and PowerShell:** pass the file name in the scan function's `outputex` parameter.
- **Jenkins:** pass the file name in the scan function's `outFilename` parameter.

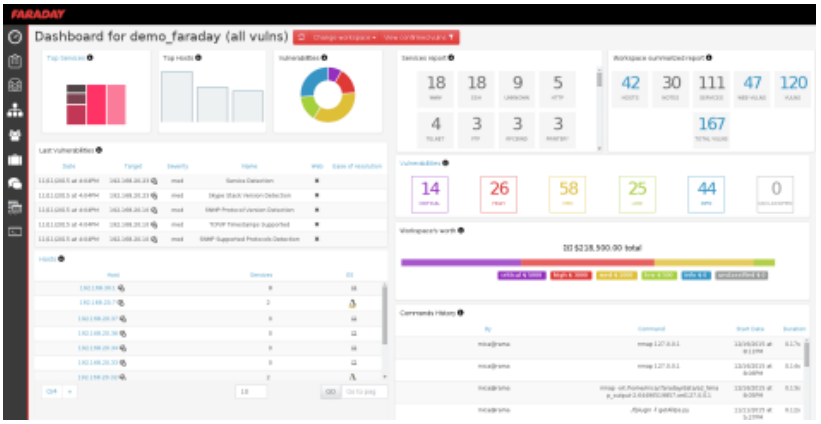
> **Note**
> DefectDojo currently documents only GitLab SAST support. DAST import may have some limitations.

Page 130 of 175

Syhunt Hybrid 7.2

Faraday

> **Applies to:** Syhunt Hybrid 7.2



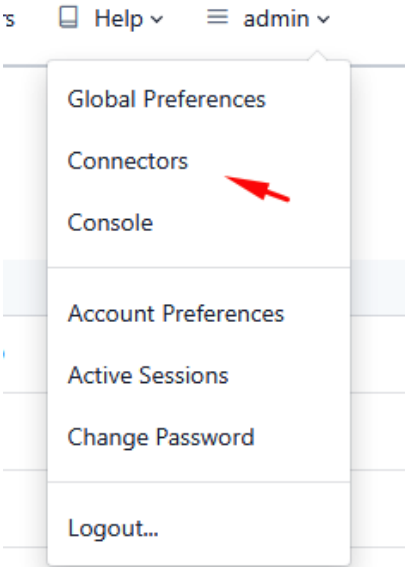
Faraday lets teams manage their attack surface from a single place while automating and speeding up key steps of vulnerability management. It brings an otherwise overwhelming number of security tools and procedures together into one simple process.

Syhunt can connect to a Faraday server's API and submit SAST or DAST results automatically. Faraday also ships with a Syhunt plugin, so Syhunt results can be uploaded through Faraday's web UI, CLI and REST API.

Submitting results automatically

1. Create a Faraday connector

1. Click **Connectors** in the Web UI's top menu.



2. Click **Add**. The **New Connector** screen opens.
3. Enter a name for the connector (for example, `MyFaraday`).
4. Set **Type** to **Faraday**.
5. Click **Create** and, after the success message, **Close**.

2. Configure it

1. Click the connector in the list. Its preferences screen opens.
2. Enter the Faraday server URL, for example `https://yourdashboard.apps.faradaysec.com/`.

Page 131 of 175

Syhunt Hybrid 7.2

3. Enter your workspace name. If you leave it empty, `my_workspace` is used.
4. Enter your username and password.
5. Click **Save**.

3. Test it

Click the connector in the list and choose **Run Test**. If everything is set up correctly, a test scan appears in your Faraday dashboard. Otherwise, an error message suggests what to fix.

4. Attach it to your scans

- **CLI:** pass `-tk:<connectorname>`.
- **Scheduled scans:** in the scheduled scan's preferences, on the **Notifications** tab, select the connector under **Issue Tracker #1** or **#2**.

Creating the connector from the CLI

```
-- Add a Faraday tracker
scancore -tracker:add
-- Enter the type (Faraday) and a name for the tracker, then press Enter.

-- Configure it
scancore -tracker:set to:mytrackername -key:api.url -v:"https://yourdashboard.apps.faradaysec.com/"
scancore -tracker:set to:mytrackername -key:auth.username.encrypted -v:yourusername
scancore -tracker:set to:mytrackername -key:auth.password.encrypted -vsecret

-- Test it
scancore -tracker:send -tid:TEST -to:mytrackername
```


Page 132 of 175

Syhunt Hybrid 7.2

Reference

Page 133 of 175

Syhunt Hybrid 7.2

Technical Specifications

> **Applies to:** Syhunt Hybrid 7.2

Contents

- A. License rights
- B. Architecture and vulnerability database
- C. Source code analysis (SAST)
- C2. Mobile and APK analysis (MAST)
- D. Dynamic analysis (DAST)
- D2. Augmented analysis (OAST)
- D3. Hybrid-augmented analysis (HAST)
- D4. API analysis (AAST)
- D5. Forensic log analysis (FAST)
- E. Management console
- F. Scan features
- G. System requirements
- H. Glossary

This document describes the technical specifications of **Syhunt Hybrid**, a highly accurate application security testing tool with deep coverage. It combines augmented HAST (hybrid application security testing), DAST (dynamic), SAST (static) and MAST (mobile) to identify vulnerabilities and weaknesses in applications.

A. License rights

- Syhunt Hybrid licenses are usually valid for **12 months**.
- Licenses include version updates and technical support, as well as technical documentation in English.
- Licenses allow an unlimited number of scans against an unlimited number of assets per year, launched from a single device (or more, if additional seats are ordered). Each license includes two Web UI users, an administrator and one other user; more can be purchased.
- Some licenses cover every supported programming language; others cover at least one. More languages can be purchased as needed.
- Licenses allow integration with AI through a configurable API endpoint. Syhunt currently supports OpenAI, DeepSeek and local models. Costs and hardware needed to run a local or cloud model are not covered by Syhunt licenses.

B. Architecture

Syhunt Hybrid is an application security assessment tool that is:

- **Hybrid:** it combines several analysis techniques (DAST, OAST, SAST, MAST and FAST, described below).
- **On-premises:** it is deployed on the customer's premises, so analyzed source code and results never leave the internal network.
- **Modular:** a single management console brings together the configuration and results of every module.
- **Asynchronous:** it runs several scans at the same time, without waiting for each one to finish.
- **Multi-process:** each scan runs as a separate operating system process.
- **Accessible from the web and the command line:** it has a responsive web interface and a CLI for launching dynamic and code scans and other tasks.

Types of testing

Type	Description
SAST	Static analysis of web application source code.

Page 134 of 175

Syhunt Hybrid 7.2

Type	Description
MAST	Static analysis of mobile application source code (Android and iOS) and Android APK files.
DAST	Dynamic analysis of web applications, with deep crawling and parameter injection.
OAST	Augmented dynamic analysis of web applications using out-of-band (OOB) techniques.
HAST	Hybrid-augmented analysis: static analysis results are automatically used to enhance augmented dynamic analysis.
AAST	Dynamic analysis of web APIs, driven by an API specification.
FAST	Forensic analysis of web server log files.

Vulnerability database

Syhunt includes an internal vulnerability database covering the public vulnerability lists and standards below. It is updated regularly, so the tool keeps up with new vulnerabilities published by international databases.

Standard	Coverage
CWE/SANS Top 25	Most Dangerous Software Errors, 2023 or later
OWASP Top 10	Web Application Security Risks, 2021 or later
OWASP Mobile Top 10	Mobile Risks, 2024 or later (PLUS)
OWASP PHP Top 5	
CWE	Common Weakness Enumeration
CVE	Common Vulnerabilities and Exposures
ISO/IEC 27001	
WASC	Web Application Security Consortium Threat Classification
WAVSEP	Web Application Vulnerability Scanner Evaluation Project
NIST SAMATE	Software Assurance Metrics and Tool Evaluation project
PCI DSS	Payment Card Industry Data Security Standard, 3.2, 3.2.1 or later

Integrations

Category	Systems
Issue tracking	Jira 7.13.0 or later GitHub GitLab
Version control	Public and private Git repositories Azure DevOps and TFS repositories (TFVC and Git) GitHub and GitLab Branches
APIs and scripting	Lua 5.1 API Web API (REST) PowerShell JSON and XML Selenium
Security dashboards	OWASP DefectDojo GitLab Security Dashboard Faraday

Page 135 of 175

Syhunt Hybrid 7.2

Category	Systems
Application vulnerability management (WAF)	F5 BIG-IP Application Security Manager (ASM) Imperva SecureSphere ModSecurity CRS OGASEC WAF
CI/CD	GitHub Actions GitLab CI and Security Dashboard Bitbucket Pipelines Jenkins Pipeline
Browsers	Syhunt Sandcat extension (Chromium-based browsers) Google Chrome Mozilla Firefox
Email	SMTP

The examples below show how to scan a Git repository through each integration:

```
-- Command prompt
scancode git://sub.domain.com/repo.git
scancode https://github.com/user/repo.git -rb:master
scancode https://dev.azure.com/user/projectname

-- GitLab CI script
- Start-CodeScan -pfcond 'fail-if:risk=mediumup' -output 'report.pdf' -outputex 'gl-sast-report.json'

-- GitHub Actions script
- Start-CodeScan -pfcond 'fail-if:risk=mediumup' -output 'report.pdf'

-- Jenkins pipeline script
syhunt.scanCode([target: 'https://github.com/someuser/somerepo.git', branch: 'master', pfcond: 'fail-
if:risk=mediumup'])

-- PowerShell
$MyScan = @{
  target = 'https://github.com/someuser/somerepo.git';
  branch = 'master';
  pfcond = 'fail-if:risk=mediumup';
  output = 'report.pdf'
}
Start-CodeScan @MyScan

-- Lua script
code:scanurl('https://github.com/someuser/somerepo.git', 'master')

-- Web API (REST, raw JSON), POST /syhunt/launch.lua
{
  "sourcetarget": "https://github.com/someuser/somerepo.git",
  "sourcebranch": "master",
  "apikey": "YOUR_API_KEY"
}
```

Supported languages and environments

Language	Environments and frameworks
C#	ASP.NET Web Forms (ASPX) ASP.NET Core MVC .NET MAUI (iOS and Android) Xamarin (iOS and Android) Blazor Razor Pages
Dart	Flutter (iOS and Android) Server side

Page 136 of 175

Syhunt Hybrid 7.2

Language	Environments and frameworks
Java	JEE JSP Android Spring Framework
JavaScript	Client side Server side Node.js (plain, Express.js and Koa.js) Angular (2 or later) AngularJS JScript (Classic ASP) Electron (desktop)
Kotlin	Ktor
Lua	ngx_lua mod_lua CGILua Lua Pages
Objective-C, C and C++	iOS
Object Pascal	Delphi Delphi XE Lazarus DWS
Perl	
PHP	
Python	CGI mod_python PSP WSGI Django
Ruby	Rails ERB mod_ruby
Swift	iOS
TypeScript	
VB	VB.NET (ASP.NET) VBScript (Classic ASP)

C. Source code analysis (SAST)

Modules	Syhunt Code; Syhunt Mobile
Description	Analyzes the source code of web and mobile applications for security vulnerabilities.
Types of analysis	SAST (Static Application Security Testing) MAST (Mobile Application Security Testing)
Target applications	Web applications (including web services) Mobile applications (Android and iOS) Desktop applications

Page 137 of 175

Syhunt Hybrid 7.2

Target languages and platforms	APK (Android) ASP.NET and Classic ASP .NET MAUI / Xamarin (iOS and Android) Blazor / Razor Pages Dart (Android and iOS) HTML Java (including Android) JavaScript Kotlin (including Android) Lua Objective-C, C and C++ (iOS) Object Pascal (Delphi) Perl PHP Python Ruby Swift (iOS) TypeScript
Git integration	Yes

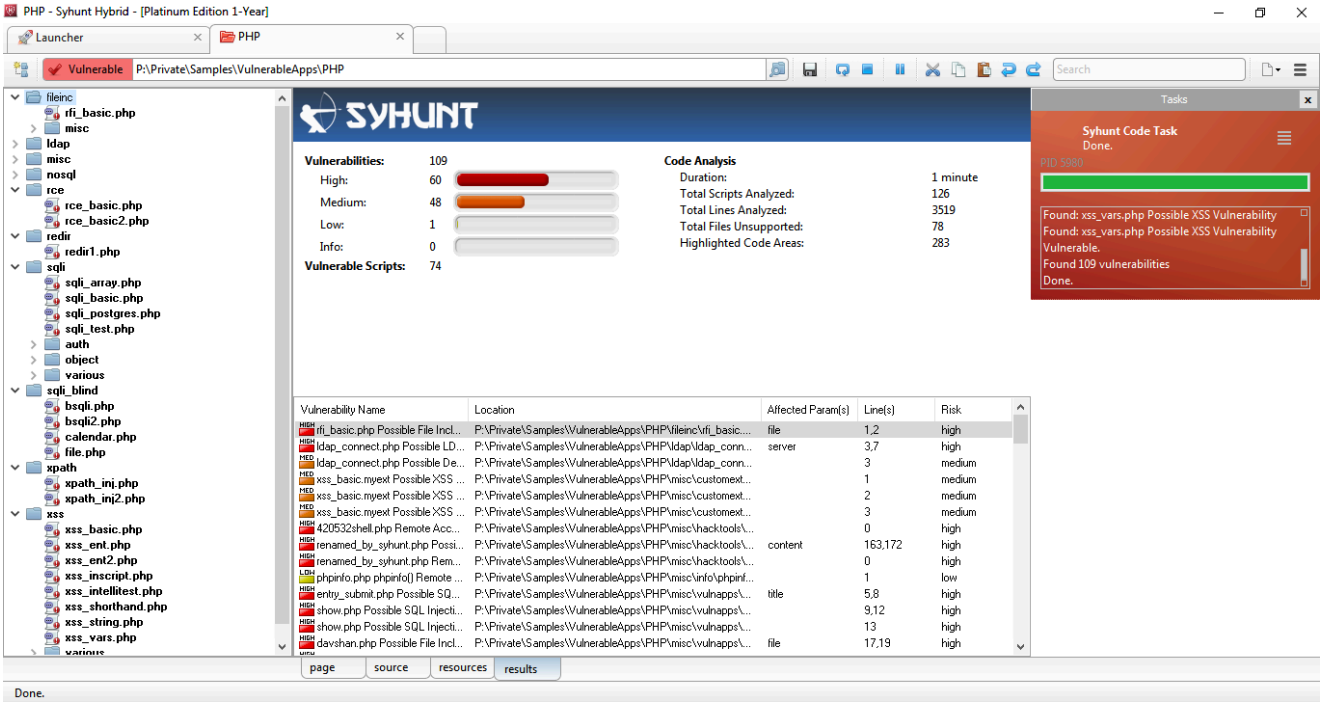
Capabilities:

- Detects known application vulnerabilities in every supported language and environment.
- Supports code embedded in HTML, print shorthands and string interpolation.
- Finds both client-side and server-side vulnerabilities.
- Detects outdated, vulnerable scripts, local or remote, such as vulnerable versions of AngularJS, jQuery, fullPage, Bootstrap and Moment.js.
- Analyzes complete codebases, code snippets and configuration files.
- Performs incremental analysis: key results and data from scans of a codebase are stored automatically and used to speed up later scans.
- Audits several programming languages in the same scan.
- Supports TypeScript, detecting vulnerabilities before the code is compiled to JavaScript (client and server side, Node.js and Angular).
- Identifies key areas of code, such as specific HTML markers, JavaScript, XHR requests, entry points and interesting keywords, and lets you navigate between them.
- Finds vulnerabilities caused by poorly designed code, that is, programming errors that expose the system to attack.
- Detects **shadow AI**, including use of DeepSeek, OpenAI, OpenAI-compatible and other emerging AI API endpoints.
- Analyzes configuration files to assess threats and identify countermeasures at the server, environment or application configuration level.
- Recognizes input filtering and validation, for accurate results free of false positives.
- *AI-powered:* adds fixed-code examples to reports, next to each vulnerable piece of code.
- *AI-powered:* adds a confidence score to each finding in reports, the estimated likelihood (as a percentage) that it is a true positive rather than a false positive.

Page 138 of 175

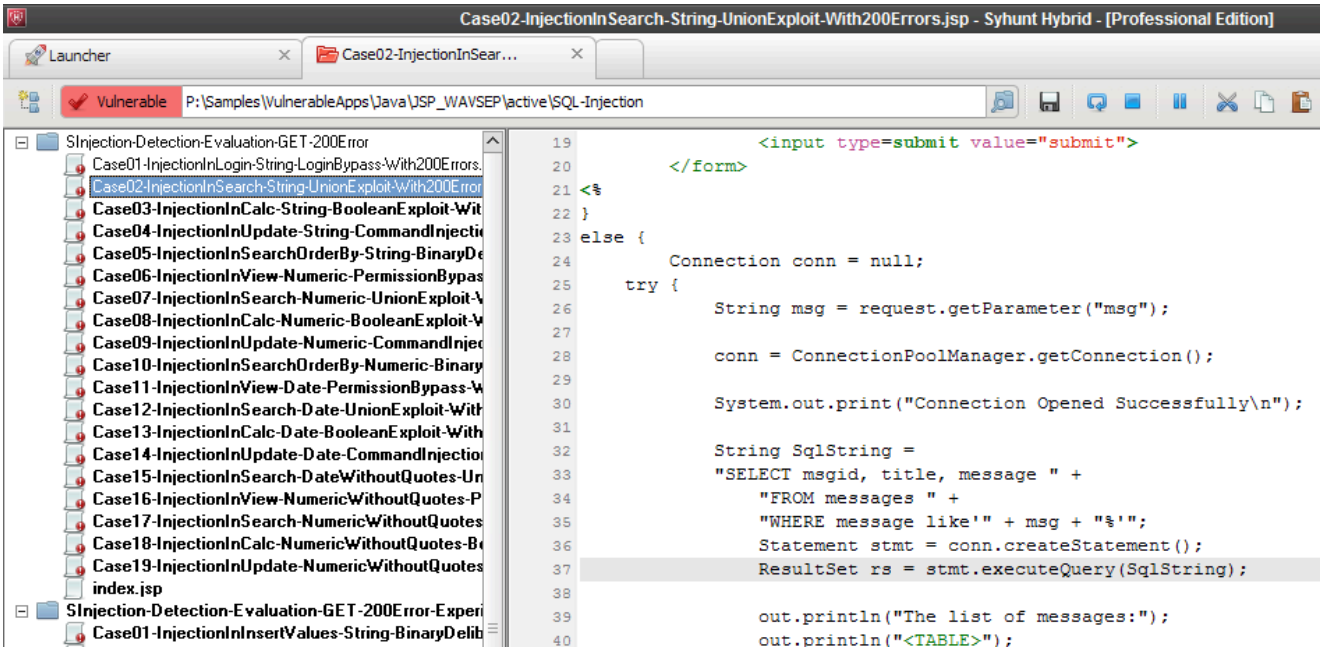
Syhunt Hybrid 7.2

Real-time scan view



The interface shows the status of running code scans in real time, including:

- the tree of analyzed and vulnerable files;
- scan duration;
- number of scripts analyzed and vulnerable;
- number of vulnerabilities found, in total and by severity;
- number of lines analyzed;
- the list of vulnerabilities found.



It also highlights the points in the code where a single fix would remediate several of the vulnerabilities found.

Page 139 of 175

Syhunt Hybrid 7.2

C2. Mobile and APK analysis (MAST)

Syhunt Code performs mobile application security testing on Android and iOS source code and on Android APK files, covering the main OWASP Mobile Top 10 and CWE/SANS Top 25 risks, such as:

1. Improper credential usage
2. Inadequate supply chain security
3. Insecure authentication/authorization
4. Insufficient input/output validation
5. Insecure communication
6. Inadequate privacy controls
7. Insufficient binary protections
8. Security misconfiguration
9. Insecure data storage
10. Insufficient cryptography

D. Dynamic analysis (DAST)

Module	Syhunt Dynamic Augmented
Description	Detects security vulnerabilities in dynamic web applications and web servers.
Types of analysis	DAST (Dynamic Application Security Testing) OAST (Out-of-Band Application Security Testing)
Target applications	Web applications and web servers
Target languages and platforms	ASP.NET and Classic ASP HTML and JavaScript Java Lua Perl PHP Python Ruby
Optimized for HTTP servers	Apache Apache Tomcat Microsoft IIS NGINX OpenResty
Protocols	HTTP 1.0 and 1.1 HTTPS (SSL 2, SSL 3, TLS 1 to TLS 1.3) IPv6 Proxies (HTTP, SOCKS 4 and SOCKS 5)
HTTP features	Keep-Alive GZIP compression Authentication: Basic, Bearer, Digest (MD5/SHA-256/SHA-512), form-based and NTLM*

* NTLM is available on Windows only. Kerberos support is planned.

Capabilities:

- Detects known web application vulnerabilities in every supported language and environment.
- Performs **augmented dynamic analysis**, combining DAST and OAST.
- Performs **hybrid-augmented analysis**, combining DAST, OAST and SAST.
- Performs incremental analysis: key results and data from scans of a target URL are stored automatically and used to speed up later scans.

Page 140 of 175

Syhunt Hybrid 7.2

- Supports a wide range of authentication methods, including automatic login-form detection (Smart and Standard), recorded browser logins (Recorder), Selenium-based logins (Selenium-Auto and Node.js scripts), HTTP authentication and manual login through the Sandcat extension.
- Reports whether an authenticated session was maintained from the start to the end of an authenticated scan.
- Detects outdated, vulnerable scripts, server software and other components.
- Finds both client-side and server-side vulnerabilities.
- Finds in-band, inferential and out-of-band (OOB) vulnerabilities.
- Detects known vulnerable applications in every supported language, including ColdFusion, Flash and Server-Side Includes (SSI).
- Detects **shadow AI** through injection techniques, including use of DeepSeek, OpenAI and other emerging AI systems.
- Automatically performs structure, login-form and HTTP authentication brute-force attacks.
- Automatically detects logout pages and avoids them during the scan, with an optional manual configuration.
- Supports both invasive and non-invasive scanning.
- Lets you turn off denial-of-service (DoS) tests that could affect the application's availability.
- Lets you configure the number of retries and the timeout for connecting to the application.
- Supports applications built on content management systems such as Drupal, Joomla, Wix and WordPress.
- Supports single-page applications (SPAs).
- Identifies the technologies used by the application and optimizes scan time accordingly.
- Identifies hidden versions of server software and components (Hunter-Sense™), such as Apache, NGINX, PHP, mod_ssl, OpenSSL and Phusion Passenger.
- Injects data and manipulates parameters in URLs and forms (GET and POST).
- Applies injection mutations to cover every supported language and platform.
- Performs time-based injection checks with a 0% false-positive rate for code injection, expression language injection, NoSQL injection, remote command execution and SQL injection.
- Finds vulnerabilities such as SQL injection, NoSQL injection, OS command injection, disclosure and code injection using inferential (response time), in-band (error message or print), out-of-band (OOB) and passive techniques.
- Includes an optional AJAX crawler for deep crawling of AJAX-heavy applications built with popular frameworks such as Angular, AngularJS and React, with pre-rendering and local storage support.

Browsing and crawling

Syhunt maps the target application's structure, including every link and entry point, while emulating a modern browser with support for:

Area	Support
Web standards	HTML5 CSS3
HTML analysis	Relative paths Standard HTML Non-standard or malformed HTML Form recognition
Browser emulation	Google Chrome Mozilla Firefox Microsoft Edge Referer submission Pre-rendering Local storage
JavaScript	JavaScript analysis and execution DOM emulation Browser behavior XHR (XMLHttpRequest) / AJAX calls External JavaScript files

Page 141 of 175

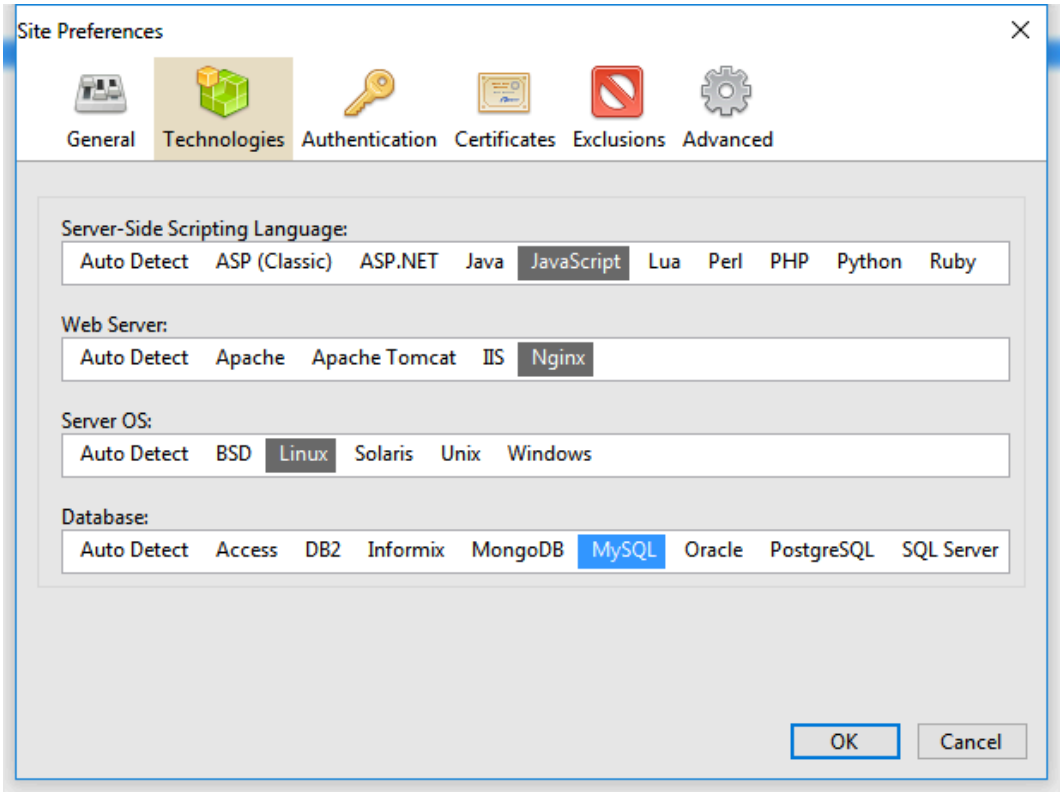
Syhunt Hybrid 7.2

Area	Support
User interaction simulation	Key presses Mouse clicks Form autofill Automatic login Accidental logout prevention
Cookies	Cookie management Session management
Redirects	HTTP Meta refresh JavaScript
Process isolation	Each scan runs as a separate process
robots.txt	Analyzed when available

Syhunt handles large, complex websites with dynamically generated content and prevents loops while mapping the application. You can also limit scan depth, including:

- the maximum number of links per server and per page;
- the maximum URL size (bytes) and HTTP response size (kilobytes).

Site preferences



Syhunt Dynamic lets you create scan profiles for assets, specifying:

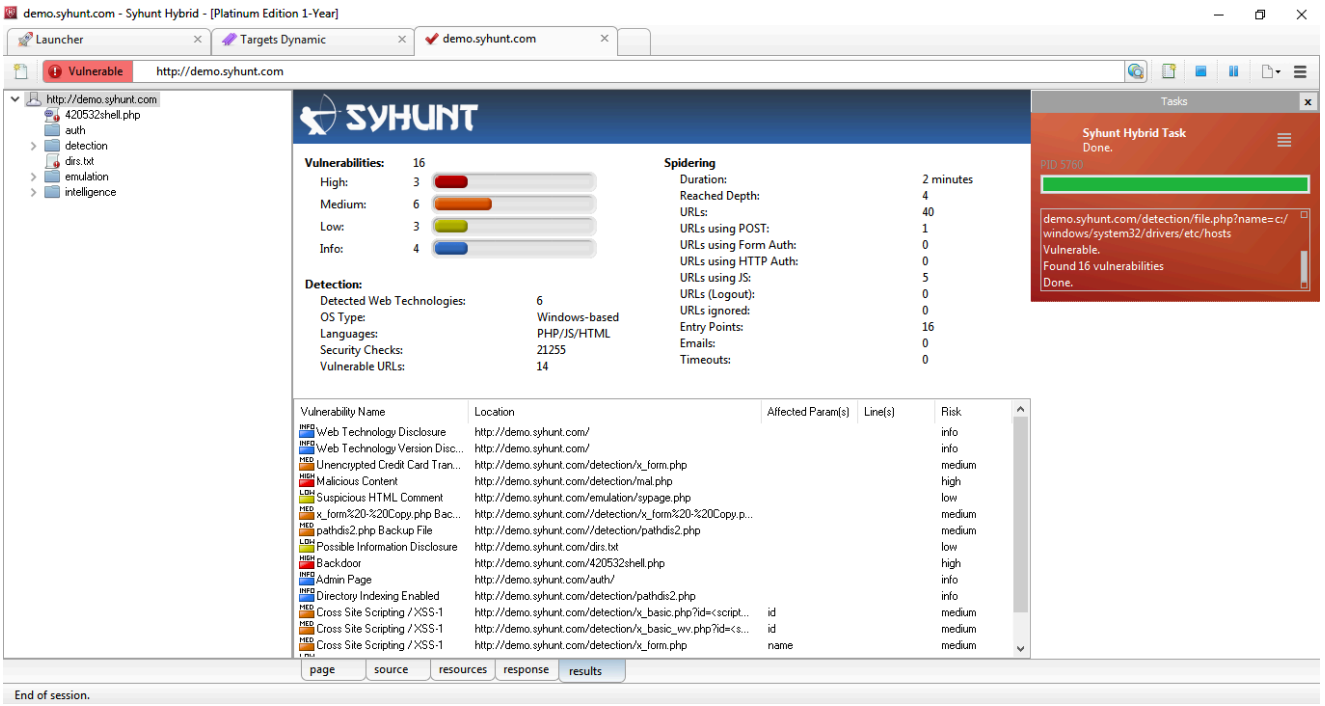
- the technologies used by the application (server-side language, web server, operating system and database), to optimize scan time;
- start URLs;
- SSL certificates and authentication credentials, to scan while logged in to the target;
- objects to exclude from the scan, such as specific paths, forms and vulnerabilities;
- depth and layer limits;
- signatures for detecting custom 404 error pages;

Page 142 of 175

Syhunt Hybrid 7.2

- manually configured cookies and session tokens.

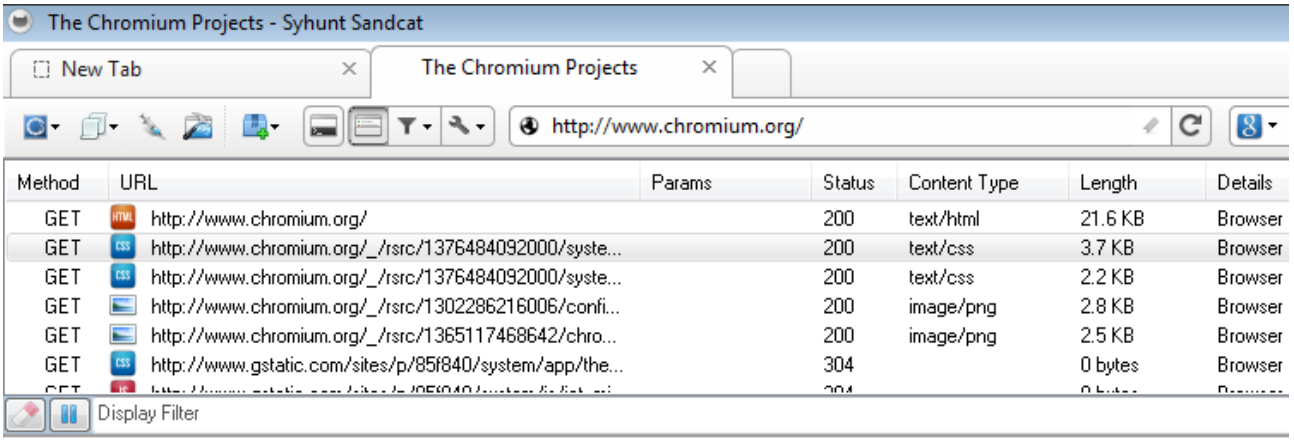
Real-time scan view



The interface shows the status of running dynamic scans in real time, including:

- the site tree of found and vulnerable paths (site map and structure);
- scan duration;
- number of vulnerable URLs;
- number of URLs using POST, authentication and JavaScript;
- number of entry points;
- number of timeouts;
- number of vulnerabilities found, in total and by severity;
- number of security checks performed;
- detected web technologies;
- the target's operating system type;
- the target's programming languages;
- the depth reached;
- the list of vulnerabilities found.

Manual and browser-assisted testing



Page 143 of 175

Syhunt Hybrid 7.2

Syhunt Dynamic supports manual and browser-assisted testing. Since version 7.2 this is done through the Syhunt Sandcat browser extension; versions 7.1 and earlier included the built-in Sandcat Browser. Capabilities include:

- manual login to web applications;
- scanning after logging in with CAPTCHAs, tokens and other two-factor authentication mechanisms;
- scanning after manual crawling/navigation;
- capturing URLs and entry points.

The classic Sandcat Browser (7.1 and earlier) also offered live HTTP headers; previews of common web file formats (CSS, Flash, HTML, BMP, GIF, ICO, JPG, PNG, SVG, JavaScript, JSON, text and XML); automatic JavaScript deobfuscation; request replay; a fuzzer; a script executor; and HTTP/XHR editors with a request loader.

D2. Augmented analysis (OAST)

Module	Syhunt Signal
Description	Detects out-of-band security vulnerabilities in dynamic web applications.
Type of analysis	OAST (Out-of-Band Application Security Testing)
Target applications	Web applications

Syhunt Dynamic Augmented works with the online **Syhunt Signal** service to perform automated out-of-band testing. Syhunt Signal:

- lets the scanner detect high-risk out-of-band (OOB) vulnerability variants that would otherwise be invisible;
- listens for requests forced from a vulnerable target server during the scan and signals them back to the scanner;
- automatically correlates the alerts it receives with the attack requests that triggered them;
- adds the OOB vulnerabilities found to the report and the user interface;
- returns zero false positives;
- automatically exfiltrates data from a vulnerable target, using environment- and OS-specific commands and techniques, and adds it to the scan results;
- requires an active Internet connection.

Out-of-band vulnerability types detected by Syhunt Signal:

- Command execution
- Remote file inclusion (RFI)
- Server-side request forgery (SSRF)
- SQL injection
- XML external entity (XXE) injection

D3. Hybrid-augmented analysis (HAST)

Syhunt Hybrid combines the results of Syhunt Dynamic, Syhunt Signal and Syhunt Code to perform **hybrid-augmented analysis** (augmented HAST), in which static analysis results are automatically used to enhance augmented dynamic analysis. It:

- combines SAST, DAST and OAST;
- performs hybrid analysis of client-side JavaScript code (SAST-in-DAST);
- detects and confirms vulnerabilities dynamically by simulating inferential, in-band and out-of-band attacks, using entry points and other information obtained from the source code.

D4. API analysis (AAST)

Module	Syhunt API

Page 144 of 175

Syhunt Hybrid 7.2

Description	Detects security vulnerabilities in web APIs.
Types of analysis	AAST (API Application Security Testing) OAST (Out-of-Band Application Security Testing)
Target applications	Web APIs
Specification formats	API Blueprint, OpenAPI 2 and 3, Swagger 1 to 3, Postman Collection 1 to 2.1, GraphQL (with REST annotations), I/O Docs, Google Discovery, RAML, WADL

An API scan starts from an API specification instead of a start URL. Syhunt maps the API from the specification, validates that the API behaves as specified, and then tests its endpoints with the same injection techniques used in dynamic scans, covering both the OWASP Top 10 and the OWASP API Security Top 10. See Web API Scanning.

D5. Forensic log analysis (FAST)

Module	Syhunt Forensic
Description	Detects attacks against dynamic web applications.
Type of analysis	FAST (Forensic Application Security Testing)
Target applications	Web applications

Syhunt Forensic performs on-demand heuristic security analysis of web server logs to detect attacks, including:

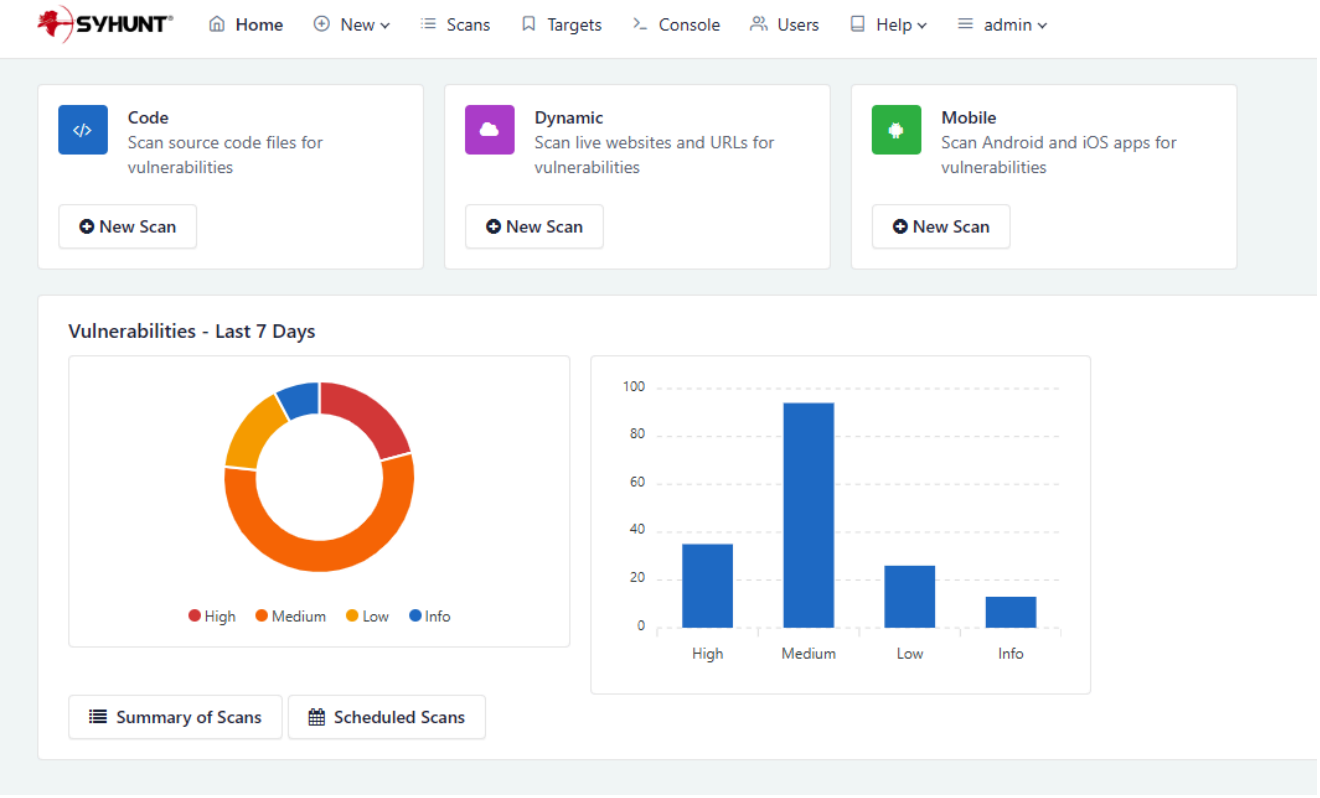
- identifying the origin (IP address), country, type and methods of attempts to compromise web applications;
- reconstructing attack sessions, accurately separating legitimate from malicious traffic, and automated from manual attacks;
- detecting intrusions, backdoor installation and the intrusion tools used;
- detecting attempts to exploit OWASP Top 10 vulnerabilities and the use of defense evasion techniques;
- supporting logs from Apache, Microsoft IIS and NGINX, with automatic format detection.

See Forensic Log Analysis.

Page 145 of 175

Syhunt Hybrid 7.2

E. Management console



The management console brings together module settings and the results of SAST, DAST and FAST scans. It:

- gives access to stored results of finished and running scans;
- can be used without prior knowledge of information security or secure coding;
- needs little or no user intervention before and during scans;
- shows the progress and risk level of each running analysis graphically;
- shows license information: license type, expiration date and supported programming languages;
- lets you disable vulnerability detection rules and see which rules are disabled;
- lets you ignore one or more specific vulnerabilities;
- lets you pause, resume and cancel scans immediately;
- lets you find, delete, export and import scan results;
- lets you add and manage assets;
- lets you export and import target lists as CSV or list files;
- lets you export and import the tool's settings;
- shows the list of vulnerabilities found in real time;
- generates summary reports of scans by scan method;
- lists all sessions for a given month;
- highlights the syntax of every supported programming language;
- *(classic interface)* includes a notepad for manual testing, with common injection string generators, hash generators, encoders and decoders, HTML functions and text manipulation;
- includes an Extension Development Kit (EDK) for adding new features.

User access control

- The Web UI enforces a strict password policy:
  - minimum lengths for admin and non-admin passwords;
  - no reuse of recent passwords;
  - complexity rules: no username or part of the user's full name; at least one uppercase letter, one lowercase letter, one digit and one non-alphanumeric character; no three or more identical characters in a row (such as `AAA`) and no three consecutive ascending or descending characters (such as `123`, `321`, `ABC`, `CBA`).

Page 146 of 175

Syhunt Hybrid 7.2

- The Web UI has built-in brute-force protection, generates session IDs securely and hashes passwords securely.
- Role-based access: Administrator, Operator, Restricted Operator and Viewer.

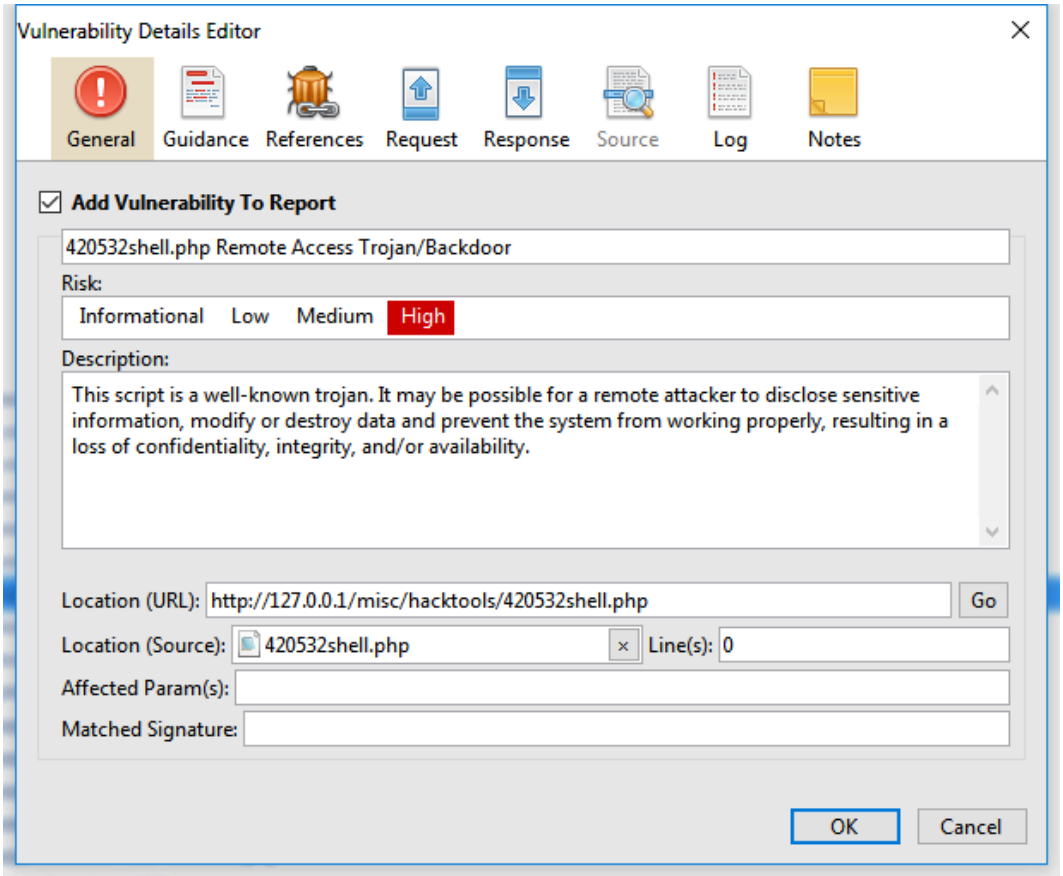
Scheduling scans

Scheduled scans support:

- time, date or day of the week;
- scan type (dynamic or code);
- scan target;
- scan time limit;
- report generation, with a choice of report template;
- emailing the report when the scan finishes;
- notifications to issue trackers or security dashboards;
- a fail condition based on minimum severity;
- hidden scans (without a window);
- exporting the command line for use in third-party schedulers and other tools.

Vulnerability alerts

The management console lets you view and edit vulnerability alerts, including changing their status and severity and adding comments.



Each alert includes:

- vulnerability name and description;
- location (a URL or file);
- references to known vulnerability databases, such as CVE, CWE, NVD and OSVDB, if any;
- CVSS score (versions 2 and 3);
- severity (High, Medium, Low or Informational);
- a unique Syhunt check ID;

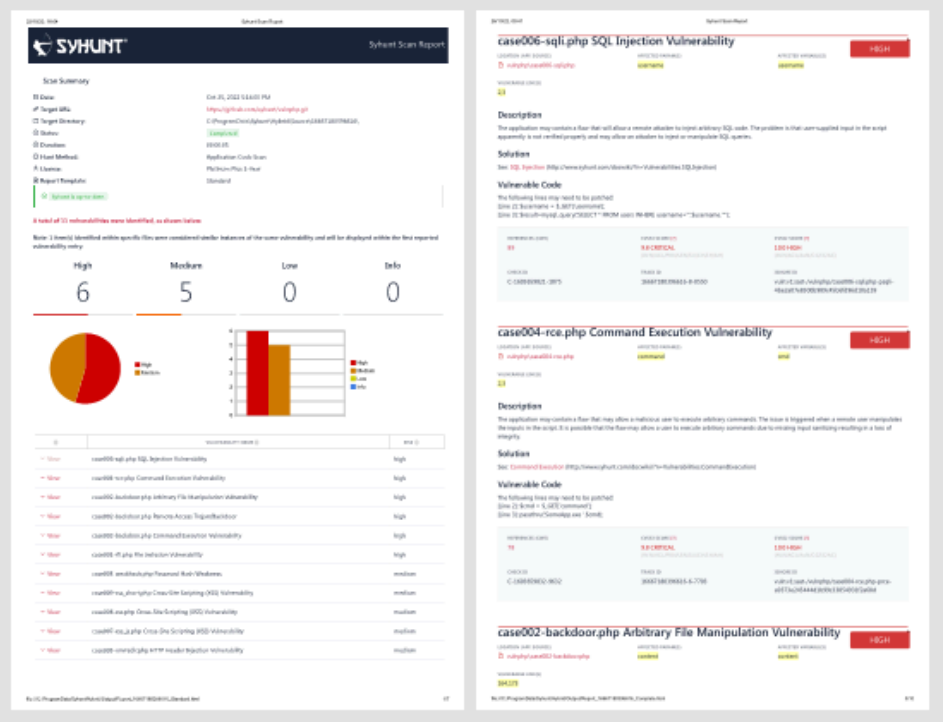
Page 147 of 175

Syhunt Hybrid 7.2

- a track ID, unique to the vulnerability in each scan;
- a remediation guide;
- affected parameters and variables;
- affected line numbers;
- a short extract of the vulnerable code, when applicable;
- remediation code examples, when applicable;
- the request, headers and HTTP response, for dynamic analysis;
- exfiltrated data, if any;
- user notes.

F. Scan features

Reporting and results



Syhunt presents scan results and generates reports on the vulnerabilities found. It:

- performs pass/fail testing based on the severity of the vulnerabilities found;
- compares two scans of the same target or codebase, showing new, unchanged and removed vulnerabilities in a report and in the management console;
- generates reports in English, German, Spanish, French, Italian, Japanese, Korean and Portuguese, with translated vulnerability descriptions and solutions;
- emails results automatically to a chosen address;
- generates an alert for each unique type of vulnerability found;
- confirms from traces whether an application has been breached;
- supports CVSS 2.0 and 3.0, including base score and other metrics, to communicate severity and help prioritize the response;
- generates reports in PDF and HTML (for reading) and XML, JSON and CSV (for other tools);
- generates responsive HTML reports for mobile, tablet and desktop;
- groups similar instances of a vulnerability into a single item;
- sorts vulnerabilities by CVSS 3, CVSS 2 (0.0 None to 10.0 Critical) or four severity levels (High, Medium, Low, Informational);
- creates custom reports with your own title, logo, notes and footer, and a choice of templates and sections;
- generates reports covering several selected sessions;
- adds screenshots of authenticated pages to reports automatically.

Page 148 of 175

Syhunt Hybrid 7.2

Report templates

Template	Contents
Standard	Standard management report.
Comparison	Compares a scan's results with previous scans, with charts and tables that make the differences clear.
Compliance	Pass/fail report against the OWASP Top 10, the CWE/SANS Top 25 Most Dangerous Software Errors and PCI DSS, in their current versions.
Compliance (Mobile)	Pass/fail report against the OWASP Mobile Top 10, the CWE/SANS Top 25 Most Dangerous Software Errors and PCI DSS, in their current versions.
Complete	Comparison, compliance and all technical and vulnerability details.

Reports include:

- general scan details: start date, target, status, duration and scan method;
- charts and statistics: total vulnerabilities and totals by severity;
- vulnerability details: everything shown for each alert in the management console;
- coverage details: the mapped structure, lists of forms, emails, JavaScript files and other resources found, and the technologies and platforms detected;
- the version and a short description of the active license.

Scan methods

Method	Focus
Application Scan (Dynamic)	Maps the website's structure and performs passive analysis and active attacks.
Application Code Scan	All types of vulnerabilities in source code.
Application Scan (Server-Side Focused)	Server-side vulnerabilities only (dynamic or code).
Spider Only	Maps the website's structure only, without passive analysis or attacks.
Passive Scan	Maps the website's structure with passive analysis, without attacks.
CWE Top 25	The CWE Top 25 Most Dangerous Software Errors.
OWASP Top 10	The OWASP Top 10 Web Application Security Risks.
PHP Top 5	The OWASP PHP Top 5, though not limited to PHP.
Fault Injection	Data injection flaws such as XSS, SQL injection, file inclusion and command execution.
Structure Brute Force	Common backup files, administration pages and similar exposures.
Backup Files	Backup, hidden and obsolete files, less aggressively than Structure Brute Force.
Complete Penetration Test	Every dynamic test, extensively; time-consuming.
SQL Injection	SQL and NoSQL injection.
XSS	Cross-site scripting and evasion of anti-XSS filters.
File Inclusion	Local and remote file inclusion (LFI and RFI).
Malicious Content	Malware, backdoors, hidden entry points and signs of intrusion.
Unvalidated Redirects	Redirect vulnerabilities.

See Hunt Methods for CLI names and details.

Page 149 of 175

Syhunt Hybrid 7.2

SQL injection detection

Syhunt finds SQL and NoSQL injection through dynamic analysis, using inferential (time-based), in-band (error-based) and out-of-band techniques, and through source code analysis:

Database	DAST (inferential)	DAST (in-band)	SAST	OAST (OOB)
NoSQL injection	✅	✅	✅	N/A
MongoDB	✅	✅	✅	N/A
SQL injection	✅	✅	✅	✅
Firebird/InterBase	N/A	✅	✅	N/A
IBM Db2	N/A	✅	✅	N/A
Informix	N/A	✅	✅	N/A
MariaDB / MySQL	✅	✅	✅	N/A
Microsoft Access	N/A	✅	✅	N/A
Microsoft SQL Server	✅	✅	✅	✅
Oracle	✅	✅	✅	✅
PostgreSQL	✅	✅	✅	N/A
SQLite	✅	✅	✅	N/A
Sybase	✅	✅	✅	N/A

Code injection and code disclosure detection

Syhunt finds code injection and code disclosure through dynamic analysis, using inferential (time-based), in-band (print-based) and passive techniques, and through source code analysis:

Language or environment	DAST (inferential)	DAST (disclosure)	DAST (in-band)	SAST (injection)
Classic ASP	✅	✅	✅	✅
ASP.NET	✅	✅	✅	✅
Java	✅	✅	✅	✅
JavaScript	✅	N/A	✅	✅
Lua (NGINX, Apache, CGILua and others)	✅	✅	✅	✅
Object Pascal (including DWS)	✅	N/A	✅	✅
Perl	✅	✅	✅	✅
PHP	✅	✅	✅	✅
Python	✅	N/A	✅	✅
Ruby	✅	N/A	✅	✅
Server-Side Includes (SSI)	N/A	✅	N/A	N/A

OS command injection detection

Syhunt finds OS command injection through dynamic analysis, using inferential (time-based), in-band (print-based) and out-of-band techniques, and through source code analysis:

Page 150 of 175

Syhunt Hybrid 7.2

OS	DAST (inferential)	DAST (in-band)	SAST	OAST (OOB)
Unix/Linux	✅	✅	✅	✅
Windows	✅	✅	✅	✅
BSD	✅	✅	✅	✅
Solaris	✅	✅	✅	✅
macOS	✅	✅	✅	✅
iOS	N/A	N/A	✅	N/A
Android	N/A	N/A	✅	N/A

Vulnerability categories

Syhunt detects the following types of vulnerabilities, weaknesses and exposures in web applications and, where applicable, mobile applications. For the detailed list of checks, see Vulnerability Checks.

- API abuse and misuse
- Arbitrary file manipulation
- Backdoors (web-based)
- Bad practices
- Broken authentication
- Broken cryptography
- Buffer overflow
- Code injection, expression language (EL) injection and regular expression injection
- Command execution
- Common backup files and folders, and backups with common or double extensions
- Common form weaknesses, including email form hijacking, hidden price fields, autocomplete enabled and unencrypted credit card transactions
- Cookie manipulation
- Cross-site scripting (XSS), including DOM-based XSS, HTML5-specific XSS, weak XSS filters and cross-frame scripting (XFS)
- Cross-site scripting (XSS) in LLM-powered web applications
- Dangerous methods
- Debug entry points, including hidden debug parameters
- Default content
- Denial of service (DoS), client and server side
- Directory listing
- Directory traversal
- Disclosure of paths, source code, databases, passwords, internal IP addresses, web technologies and more
- Forgery: log forging, cross-site request forgery (CSRF) and server-side request forgery (SSRF)
- Hardcoding or logging of sensitive information
- HTTP header injection and HTTP response splitting
- Inappropriate or malicious content
- Local or remote file inclusion
- Information leakage
- LDAP injection
- JSON, XML, XPath and XXE (XML external entity) injection
- NoSQL, SQL and HQL injection
- SSI (Server-Side Includes) injection
- Insecure communication
- Insecure cryptographic and hash algorithms
- Insecure data storage: missing or insufficient data protection
- Insecure randomness
- Insecure salting
- Missing or weak HTTP security headers

Page 151 of 175

Syhunt Hybrid 7.2

- Security misconfiguration
- Sensitive information on the client side
- Suspicious comments in source code and HTML
- Uncontrolled format strings
- Unencrypted login
- Unvalidated redirects
- Use of local storage, and confidential data stored in local storage
- Weak password hashing
- Weak protocols

G. System requirements

Syhunt Hybrid runs on 64-bit Windows, macOS (Apple Silicon) and Linux, and can analyze applications built for any platform, including Android, iOS, macOS, BSD, Linux, Windows, Solaris and Unix, regardless of the platform it runs on.

Requirement	Details
Memory	4 GB of available RAM (8 GB recommended)[1]
Disk	2 GB of free disk space[2]
Internet connection	Recommended for code and dynamic scans, and required by some features (see Internet connection requirements)
Operating system (64-bit)	Windows 10 or 11, or Windows Server 2016 to 2025 (x64 or ARM64) macOS Sequoia 15.4.1 or later (Apple Silicon) Linux: Kali 2024.3 or later, Ubuntu Server/Desktop 18 or later, Debian 9 or later Unofficially supported systems[3], such as the Linux distributions listed below
Git	Optional, for Git repository scans: Git on Linux/macOS, Git for Windows on Windows
Java	Java or Java Headless, on Linux and macOS
Wine	Wine64 Stable, where native binaries aren't yet available for your OS or distribution
AI	Optional, for AI-powered features: an OpenAI or DeepSeek API key, or a local AI model

[1] Not including additional RAM that may be needed for concurrent scans.
[2] Not including space for scan session data, which depends on the website or code analyzed and how often you scan.
[3] *Unofficially supported* means the product has been successfully tested and the installation documented, but Syhunt does not provide technical support for issues specific to that OS. You may encounter compatibility issues, errors or bugs; for the best performance and compatibility, use an officially supported OS.

Compatible Linux distributions

Status	Distribution
✅ Officially supported	Ubuntu Server/Desktop 18.10 and later Debian 9 and later Kali Linux 2024.3 and later
✅ Unofficially supported (successfully tested)	Fedora 32 and later CentOS 7.7 and later (Minimal or Everything) Kali Linux 2019 and later Parrot OS 4.1, 4.7 and later Linux Mint 19.2 and later openSUSE Leap 15.1 and later MX Linux 19.1 and later KDE neon 2020.03 and later Deepin 15.9 Manjaro 19 Arch Linux 2019 and later

Page 152 of 175

Syhunt Hybrid 7.2

Status	Distribution
❌ Unsupported	elementary OS 5.1 (successfully tested) and 5.0 CentOS 6.1 (successfully tested) Solus 4.1 (unstable)

Installation steps for each distribution are in the Linux installation guide.

H. Glossary

Term	Definition
Active attacks	Security analysis that performs attacks such as brute force, injection and denial of service.
AI-powered	A feature, or feature extension, powered by artificial intelligence. Available only when an integration with a supported AI system (OpenAI, DeepSeek or a local model) is configured.
Asset	A URL, source code file or repository that can be analyzed.
Augmented dynamic analysis	The combination of DAST and OAST.
DAST	Dynamic Application Security Testing: the tool interacts with a running web application to find vulnerabilities and weaknesses. Also known as black-box testing.
DOM	Document Object Model.
FAST	Forensic Application Security Testing.
HAST	Hybrid Application Security Testing: the combination of SAST and DAST.
Hybrid-augmented analysis	The combination of SAST, DAST and OAST.
Injection	Submitting data to an application's entry points and analyzing the response to determine whether its code is vulnerable.
Lua	A lightweight, extensible scripting language created in Brazil, designed for extending applications.
MAST	Mobile Application Security Testing: the tool analyzes an application's source code or package to find programming errors and conditions that indicate vulnerabilities.
NVD	National Vulnerability Database, the U.S. government's vulnerability database.
OAST	Out-of-Band Application Security Testing: the tool tries to make a web application connect to other servers, revealing vulnerabilities that would otherwise be invisible.
OOB	Out-of-band.
OSVDB	Open Source Vulnerability Database.
OWASP	Open Worldwide Application Security Project.
Passive analysis	Security analysis that finds vulnerabilities and exposures without performing attacks or drawing attention.
PLUS	A feature available only with the Hybrid Platinum Plus license.
Proof of concept (PoC)	A set of actions demonstrating that a product works as intended.
Rule	An option that enables or disables one or more vulnerability checks.
SAST	Static Application Security Testing: the tool analyzes an application's source code to find programming errors and conditions that indicate vulnerabilities. Also known as white-box testing.
XHR	XMLHttpRequest, an API for sending HTTP or HTTPS requests from JavaScript.

Page 153 of 175

Syhunt Hybrid 7.2

Preferences

This page lists the main Syhunt Hybrid preferences you can set from the command line. Preferences are either **global** (they affect every scan and target) or **site-specific** (they apply to one web application URL).

> **Note**
> Changes don't affect scans already running; they apply to scans started afterwards. Preferences set with `scancore` are persistent and affect all future scans.

Contents

- Setting values from the CLI
- Site preferences
- Dynamic preferences
- Code preferences
- Forensic preferences
- Other Hybrid preferences
- Tracker preferences

Setting values from the CLI

With `scancore -prefset` and `scancore -tracker:set`, set the value according to the preference's type:

Type	How to set it
boolean	`-v:true` or `-v:false`
string	`-v:value`, `-v:"my value"`, or `-vsecret` (for passwords)
integer	`-v:1000`
stringlist	`-fromfile:mylistfile.lst` to read the values from a file; `-v:[string1],[string2],[etc]` to list several strings; or `-v:value` / `-v:"my value"` for a single line

Setting a value of the wrong type fails with an error, which protects preferences from accidental changes:

```
scancore -prefset:hybrid.report.company.logo.urldark -v:true
Error: Invalid type (provided boolean, must be string)

scancore -tg:http://127.0.0.1 -prefset:enabled -v:1
Error: Invalid type (provided integer, must be boolean)
```

Some details:

- **String lists:** in `-v:[value1],[value2],[value3]`, each bracketed value becomes a separate line, so this is stored as:

  ```
  value1
  value2
  value3
  ```

- **Spaces:** wrap values that contain spaces in double quotes, for example `-v:"my value"` or `-v:"[my value1],[my value2]"`. Don't use single quotes.
- **Secrets:** `-vsecret` asks for the value twice, masking what you type, so sensitive values such as passwords are confirmed and never appear on the command line.
- **Files:** on Windows, the file passed to `-fromfile` can be in the current folder or given with its full path. On Linux, always give the full path, for example `-fromfile:/path/to/mylistfile.lst`.

Page 154 of 175

Syhunt Hybrid 7.2

Use `-prefprint` to display the current value of any preference. For examples of setting global and site-specific preferences, see Command Line: modifying preferences.

Site preferences

Site-specific preferences are set with `scancore -prefset` plus the `-tg` parameter, which names the target (examples). They apply only to that target, but to every DAST scan of it, however it is started: the `scanurl` command, the Web UI, the REST API, the Lua API, the PowerShell module or the classic interface.

How targets are distinguished:

- **Different paths are different targets.** For example, `scancore -tg:http://127.0.0.1 -prefset:enabled -v:true` and `scancore -tg:http://127.0.0.1/demo/ -prefset:enabled -v:true` are stored separately, even though the base URL is the same. When you run `scanurl <url>`, the preferences that match that URL's path are used.
- **Different protocols and ports are different targets.** `https://example.com`, `http://example.com` and `http://example.com:8080` each have their own preferences.
- **Need several profiles for the same URL?** Append a marker parameter, such as `?syhunt_ref=1`, `?syhunt_ref=2` and so on. The scanned application ignores it, but Syhunt treats each variant as a separate target.

Preference	Type	Description
`enabled`	boolean	**true** (default): Enables site preferences for the target URL. This is a master switch. **false**: Disables all site preferences, and only use default settings.
`name`	string	Descriptive name of the target website (for example, My company website). This name will be displayed in the user interface of Syhunt and has no impact on scans.
`critlevel`	string	Criticality Level. Valid values include: **undefined** (Not Defined), **none** (Non-Critical), **low**, **medium**, **high**, **critical**. This information will be displayed in the user interface of Syhunt and has no impact on scans. Default: **undefined**.
`dynamic.augmented.detectoob`	boolean	**true** (default): Enables the execution of OAST with Syhunt Signal (Recommended). This allows Syhunt to detect out-of-band vulnerabilities. **false**: Disables all OAST testing.
`dynamic.augmented.scanclisource`	boolean	**true** (default): Enables the execution of SAST against client-side JavaScript with Syhunt Code **false**: Disables all SAST testing over DAST.
`dynamic.advanced.performdos`	boolean	**true** (default): Enables the execution of DoS (Denial of Service) tests against the target web application. DoS attacks aim to disrupt the normal functioning of a web application by exploiting vulnerabilities or overwhelming its resources, thereby rendering it unavailable to legitimate users. **false**: Disables all DoS testing.
`dynamic.advanced.performauthforce`	boolean	**true** (default): Perform authentication brute force. If enabled, Syhunt will automatically try commonly used username and passwords against all areas of the web application that ask for credentials. **false**: Disables authentication brute forcing.
`dynamic.advanced.forcewebrootstruct`	boolean	**true** (default): Always target web root during structure brute forcing. **false**: Disables targeting of web root during structure brute forcing. This means that if the target URL is `https://example.com/demo/` brute forcing will target `https://example.com/demo/*` but not `https://example.com/*`
`dynamic.advanced.incrementalscan`	string	Incremental Scan. Accepted string values: **auto** (default) or **disabled**. Disabling incremental scanning will result in slower future scans because the scanner won't utilize data from previous scans to expedite subsequent ones.

Web technologies

Syhunt Dynamic can run tests tailored to the technologies used by a web application, which speeds up testing. By default it detects these technologies automatically, which is why most of these options default to `auto`.

Page 155 of 175

Preference	Type	Description
`dynamic.webtech.slang`	string	The key server-side scripting language of the targeted web application. Learning in advance the programming language allows Syhunt to optimize the scan and perform only checks that are relevant to the language used by the target application. The default value is: **auto**.

Accepted values for `dynamic.webtech.slang`:

Value	Meaning
`auto`	Syhunt will automatically determine the language used by the target using its fingerprint capabilities.
`lang.asp`	The target application uses classic ASP.
`lang.aspx`	The target application uses ASP.NET.
`lang.java`	The target application uses Java or JSP (Java Server Pages).
`lang.kotlin`	The target application uses Kotlin.
`lang.ssjs`	The target application uses server-side JavaScript, such as NodeJS or similar.
`lang.lua`	The target application uses Lua or LuaJIT.
`lang.perl`	The target application uses Perl.
`lang.php`	The target application uses PHP.
`lang.python`	The target application uses Python.
`lang.ruby`	The target application uses Ruby or RoR.

Preference	Type	Description
`dynamic.webtech.server`	string	The web server software of the targeted web application. Learning in advance the server software allows Syhunt to optimize the scan and perform only checks that are relevant to the server used by the target application. The default value is: **auto**.

Accepted values for `dynamic.webtech.server`:

Value	Meaning
`auto`	Syhunt will automatically determine the server software used by the target using its fingerprint capabilities.
`server.apache`	The target application uses Apache.
`server.tomcat`	The target application uses Apache Tomcat.
`server.iis`	The target application uses Microsoft IIS.
`server.nginx`	The target application uses Nginx or OpenResty.

Preference	Type	Description
`dynamic.webtech.os`	string	The operational system of the targeted web application. Learning in advance the OS allows Syhunt to optimize the scan and perform only checks that are relevant to the OS used by the target application. The default value is: **auto**.

Accepted values for `dynamic.webtech.os`:

Value	Meaning
`auto`	Syhunt will automatically determine the OS of the target using its fingerprint capabilities.
`os.bsd`	The target application runs on top of BSD.
`os.linux`	The target application runs on top of a Linux distribution.

Syhunt Hybrid 7.2

Value	Meaning
`os.solaris`	The target application runs on top of Solaris.
`os.unix`	The target application runs on top of Unix.
`os.windows`	The target application runs on top of Windows.

Preference	Type	Description
`dynamic.webtech.datsql`	string	The key SQL or NoSQL database used by the targeted web application. Learning in advance the database allows Syhunt to optimize the scan and perform only checks that are relevant to the database used by the target application. The default value is: **auto**.

Accepted values for `dynamic.webtech.datsql`:

Value	Meaning
`auto`	Syhunt will automatically determine the database of the target using its fingerprint capabilities.
`db.sql.access`	The target application uses Microsoft Access database.
`db.sql.db2`	The target application uses DB2.
`db.sql.informix`	The target application uses Informix.
`db.nosql.mongodb`	The target application uses MongoDB.
`db.sql.mysql`	The target application uses MariaDB or MySQL.
`db.sql.oracle`	The target application uses Oracle database.
`db.sql.postgresql`	The target application uses PortgreSQL.
`db.sql.sqlserver`	The target application uses Microsoft SQL Server.
`db.sql.sqlite`	The target application uses SQLite database.

Preference	Type	Description
`dynamic.webtech.haswaf`	string	The status of WAF presence in the targeted web application. The default value is: **auto**.

Accepted values for `dynamic.webtech.haswaf`:

Value	Meaning
`auto`	Syhunt will automatically determine WAF presence.
`inactive`	No WAF is present.
`active`	WAF is present.
`ipbypass`	WAF is present with IP bypass rule.

Preference	Type	Description
`dynamic.webtech.waf`	string	The manufacturer of the active WAF. The default value is: **auto**.

Accepted values for `dynamic.webtech.waf`:

Value	Meaning
`auto`	Syhunt will attempt to automatically determine the WAF manufacturer.
`waf.akamai`	Akamai

Page 157 of 175

Syhunt Hybrid 7.2

Value	Meaning
`waf.aws`	AWS
`waf.barracuda`	Barracuda
`waf.cloudflare`	CloudFlare
`waf.f5`	F5
`waf.fortinet`	Fortinet
`waf.imperva`	Imperva
`waf.msazureappgat`	Microsoft Azure Application Gateway
`waf.modsecurity`	ModSecurity
`waf.nginx`	Nginx
`waf.ogasec`	OGASEC
`waf.radware`	Radware
`waf.sucuri`	Sucuri
`waf.unknown`	Other

Technology version guessing

Syhunt Dynamic can also infer which technologies and component versions a web server is running, by analyzing various aspects of the application and its responses. This lets it learn the technology stack without any input from you, for more thorough testing.

Preference	Type	Description
`dynamic.webtech.guess.enabled`	boolean	**true** (default): Attempt to sense web server software versions using Syhunt HunterSense, even if banner and signatures are hidden (Recommended). This is a master switch. **false**: Disables guessing of web server software.
`dynamic.webtech.guess.apache`	boolean	**true** (default): Sense Apache version **false**: Disables guessing of Apache version.
`dynamic.webtech.guess.nginx`	boolean	**true** (default): Sense Nginx version **false**: Disables guessing of Nginx version.
`dynamic.webtech.guess.php`	boolean	**true** (default): Sense PHP version **false**: Disables guessing of PHP version.
`dynamic.webtech.guess.phusion`	boolean	**true** (default): Sense Phusion Passenger version **false**: Disables guessing of Phusion Passenger version
`dynamic.webtech.guess.modssl`	boolean	**true** (default): Sense ModSSL version **false**: Disables guessing of ModSSL version
`dynamic.webtech.guess.openssl`	boolean	**true** (default): Sense OpenSSL version **false**: Disables guessing of OpenSSL version

Crawling reach

These options restrict how far Syhunt Dynamic's crawler explores. By default there are no crawling limits, so the scanner can explore a site extensively; it still applies built-in rules (Syhunt calls them *optimizations*) that guide it through the site's structure and content and speed up mapping.

Page 158 of 175

Syhunt Hybrid 7.2

Preference	Type	Description
`dynamic.crawling.limitdepth`	boolean	**true**: Use depth limit. This is a master switch. **false** (default): Disables depth limit.
`dynamic.crawling.maxdepth`	integer	Maximum depth (requires use depth limit enabled). Default: **0**., which means no limit.
`dynamic.crawling.limittobase`	boolean	**true**: Limit crawling to start URL path and specific allowed URLs. This is a master switch. **false** (default): Disables crawling limitation to start URL path and specific allowed paths. If you enable this option, and for example, you are targeting `https://example.com/demo/` and the application links to `https://example.com/test/`, /test/ will be ignored. Because web application paths are often intertwined, this would restrict too much the scan, and because of that, this option is disabled by default.
`dynamic.lists.ignore.allowedpaths`	stringlist	Paths allowed to crawl (for example, /store/login.php). Default: empty list.

The following lists don't require any master key to be enabled:

Preference	Type	Description
`dynamic.lists.starturls`	stringlist	Additional Start URLs (for example, /url1.php,/url2.php). By default, Syhunt Dynamic will use the path that is part of the target URL you provide as the starting point for scanning. However, this key allows you to include additional start paths for scanning. For instance, if your target is `https://example.com/demo/` and you also want to include **/demov2/** and **/demov3/** for scanning, even if they are not directly linked within **/demo/**, adding these paths to this list ensures they will also be scanned. Default: empty list.
`dynamic.lists.ignore.urls`	stringlist	Paths to ignore (for example, /about1.php,/about2.php). Default: empty list.
`dynamic.lists.ignore.logoutpaths`	stringlist	Logout paths to ignore (for example, /logout.php). Default: empty list. Syhunt Dynamic already comes pre-configured to detect the most common logout paths.
`dynamic.lists.ignore.form.names`	stringlist	Forms to ignore by name (for example, search1,search2). Default: empty list.

Authentication

These options enable web form authentication, server authentication methods such as Basic and Bearer, or both. See Authenticated Scanning.

Preference	Type	Description
`dynamic.formauth.type`	string	Form Authentication Type. Accepted string values are: **None**, **Standard**, **Smart**, **Recorder**, **Selenium-Auto**, **Selenium-JS**, **Basic**, **Bearer**, **Digest** and **NTLM**. The default value is Standard, which means form authentication is enabled. Form authentication remains enabled as long as both the form username and password fields are not empty.
`dynamic.formauth.username`	string	Form username. Default: empty.
`dynamic.formauth.password`	string	Form password. Default: empty.
`dynamic.formauth.token.encrypted`	string	Token for Bearer authentication. Default: empty.
`dynamic.formauth.script.recorderjson.encrypted`	stringlist	Recorder JSON for authentication. Default: empty. This JSON will only be used if form authentication type is set to Recorder

Page 159 of 175

Syhunt Hybrid 7.2

Preference	Type	Description
`dynamic.formauth.script.seleniumjs.encrypted`	stringlist	Selenium Node.JS script for authentication. Default: empty. This script will only be used if form authentication type is set to Selenium-JS
`dynamic.lists.cookies`	stringlist	Session Cookies (for example, ck1=value;ck=value). Default: empty.
`dynamic.lists.cookiesforced`	stringlist	Forced Cookies (for example, ck1=value;ck=value). Default: empty.

Vulnerability ignore list

Preference	Type	Description
`dynamic.lists.ignore.vulnsbyrules`	stringlist	Vulnerabilities to ignore (by id or rule) (for example, vuln:v1:dast-my.js-xss-748). Default: empty.

Dynamic preferences

Dynamic preferences are global: they apply to every DAST scan, however it is started (the `scanurl` command, the Web UI, the REST API, the Lua API, the PowerShell module or the classic interface).

Preference	Type	Description
`dynamic.checks.dos`	boolean	**true** (default): Perform DoS (Denial of Service) tests. **false**: Disables DoS tests.

Browser emulation

Preference	Type	Description
`dynamic.emulation.redirect.autofollowinstarturl`	boolean	**true** (default): Auto handle off-domain redirect in Start URL. **false**: Disables off-domain redirect rendering.
`dynamic.emulation.mode`	string	Browser Emulation Mode. Valid options: **Chrome**, **Edge**, **Firefox**, **MSIE**, **Opera**, **Safari**. The default value is Chrome.
`dynamic.emulation.javascript.execution`	boolean	**true** (default): Enables JavaScript emulation. **false**: Fully disables JavaScript execution and emulation.
`dynamic.emulation.javascript.simuser`	boolean	**true** (default): Simulate user interaction (key press, mouse click, etc). **false**: Fully disables user interaction simulation.
`dynamic.emulation.doxhrcalls`	boolean	**true** (default): Perform XHR calls. **false**: Fully disables support for XHR calls.
`dynamic.emulation.javascript.analyzejs`	boolean	**true** (default): Analyze JavaScript code. **false**: Disables the analysis of JS code.
`dynamic.emulation.javascript.analyzexhr`	boolean	**true** (default): Analyze XHR calls. **false**: Disables the analysis of XHR calls.
`dynamic.emulation.cookies.accept`	boolean	**true** (default): Accept cookies. **false**: Fully disables cookie support.
`dynamic.emulation.cookies.maxsize`	integer	Maximum cookie size (kb). Default: **4**., which translates to 4kb.
`dynamic.emulation.cookies.maxnumber`	integer	Maximum number of cookies per site. Default: **50**.

Page 160 of 175

Syhunt Hybrid 7.2

Preference	Type	Description
`dynamic.emulation.autofill.forms`	boolean	**true** (default): Automatically fill web forms (name, address, phone, email, etc). **false**: Disables the auto-fill of forms.
`dynamic.emulation.autofill.loginforms`	boolean	**true** (default): Auto fill login web forms. **false**: Disables the auto-fill of login forms.
`dynamic.emulation.redirect.autofollow`	boolean	**true** (default): Automatically follow redirects. **false**: Disables auto follow redirects.
`dynamic.emulation.intelparser`	boolean	**true** (default): Use intelligent HTML parsing (Handles malformed HTML). **false**: Disables intelligent HTML parsing.
`dynamic.emulation.usenavbehavior`	boolean	**true** (default): Use browser behavior. **false**: Disables browser behavior.
`dynamic.emulation.referer.send`	boolean	**true** (default): Send Referer. **false**: Disables sending of referer.
`dynamic.emulation.useragent`	string	User Agent. The default value is a Chrome browser user agent.
`dynamic.emulation.forceuseragent`	boolean	**true** (default): Force the provided user agent in all situations. **false**: Disables forcing the provided user agent in all situations.

Crawling reach and time limit

These global settings limit how far the crawler goes and how long each scan can run.

Preference	Type	Description
`dynamic.crawling.max.linkspersite`	integer	Maximum number of links per server. Default: **10000**.
`dynamic.crawling.max.linksnumber`	integer	Maximum links per page. Default: **250**.
`dynamic.crawling.max.urlsize`	integer	Maximum URL length (bytes). Default: **16384**.
`dynamic.crawling.max.filesize`	integer	Maximum response size (kb). Default: **1024**.
`dynamic.crawling.analyze.robots`	boolean	**true** (default): Analyze robots.txt (if available) **false**: Disables analysis of robots.txt file.
`dynamic.options.logout.prevent`	boolean	**true** (default): Enabling this option prevents Syhunt Dynamic from inadvertently visiting the logout page of the web application during crawling or testing. This helps avoid unintentional session termination. **false**: Disables logout prevention mechanisms.
`dynamic.options.logout.detect`	boolean	**true** (default): Relogin after premature session termination when needed. **false**: Disables relogin.

Preference	Type	Description
`syhunt.dynamic.options.timelimit.enabled`	boolean	**false** (default): Do not use any time limit. **true**: Enable time limit. This is a master switch
`syhunt.dynamic.options.timelimit.value`	integer	The scan time limit parameter determines the maximum duration for a scan. Default: empty., indicating no limit. Accepted time limits include formats like: **1d** for one day, **3h** for three hours, **2h30m** for two hours and thirty minutes, or **50m** for fifty minutes. Once this maximum duration is reached, the scan automatically cancels itself

Page 161 of 175

Syhunt Hybrid 7.2

Delay

Sets a delay, in milliseconds, between requests. We advise against it for DAST, which typically involves millions of requests: any delay can make scans take much longer.

Preference	Type	Description
`dynamic.options.delay.enabled`	boolean	**false** (default): Do not use delay between requests **true**: Enables delay between requests. This is a master switch.
`dynamic.options.delay.value`	integer	Delay (ms). Default: **3000**.
`dynamic.options.delay.userandom`	boolean	**false** (default): Do not use random delay. **true**: Enables random delay between requests

Evasion

Preference	Type	Description
`dynamic.evasion.detection.honeypot`	boolean	**true** (default): Detect if the target application is a honeypot. **false**: Disables detection of honeypot.
`dynamic.evasion.detection.waf`	boolean	**true** (default): Attempts to detect if the application is behind WAF (web application firewall) **false**: Disables detection of WAF.
`dynamic.evasion.reqdistrand`	boolean	**true**: Distribute requests randomly during crawling (experimental) **false** (default): Disables random distribution of requests.
`dynamic.evasion.evadeflt`	boolean	**true** (default): Use filter evasion **false**: Disables filter evasion. This is a master switch.
`dynamic.checks.sub.xssev`	boolean	**true** (default): Evasion Technique: Common XSS filter evasion techniques. **false**: Disables Common XSS filter evasion techniques.
`dynamic.checks.sub.utf8dev`	boolean	**true** (default): Evasion Technique: UTF-8 Decode. **false**: Disables UTF-8 Decode evasion technique.

Protocol

The defaults below are recommended: they are optimized for handling web traffic efficiently and give the best scan performance.

Preference	Type	Description
`dynamic.protocol.version`	string	Protocol Version. Accepts: HTTP/1.1 and HTTP/1.0. Default: **HTTP/1.1**. This is also the recommended value.
`dynamic.protocol.ssl.type`	string	SSL Type. Accepts: all, SSLv2, SSLv3, TLSv1, TLSv1_1, TLSv1_2, TLSv1_3, SSHv2. Default: **all**. This is also the recommended value, since it works with all supported SSL types instead of restricting to a specific SSL type.
`dynamic.protocol.keepalive`	boolean	**true** (default): Enable Keep-Alive support (recommended). **false**: Disables Keep-Alive support. This negatively impacts the scan speed since Keep-Alive speeds up connections.
`dynamic.protocol.usegzip`	boolean	**true** (default): Enable GZIP compression support (recommended). **false**: Disables GZIP compression support. This negatively impacts the scan speed since GZIP compression speeds up connections.
`dynamic.protocol.mknocache`	boolean	**true** (default): Make no-cache requests. **false**: Disables no-cache requests.
`dynamic.protocol.timeout.value`	integer	Timeout (ms). Default: **10000**.

Page 162 of 175

Syhunt Hybrid 7.2

Preference	Type	Description
`dynamic.protocol.retries`	integer	Number of retries after timeout. Default: **2**.
`syhunt.dynamic.options.augmented.jndiserver`	string	JNDIExploit Server (Host without port) for Log4Shell detection. Default: empty.
`dynamic.lists.protocol.customhdr`	stringlist	Custom Request Headers (for example, X-Custom:AValue). Default: empty.

Vulnerability ignore list

Each DAST vulnerability entry in a report ends with its ignore ID, the simplest and recommended way to suppress a finding. For broader scenarios, use ignore rules.

Preference	Type	Description
`dynamic.lists.ignore.vulnsbyrules`	stringlist	Vulnerabilities to ignore (by rule or id). Default: empty list.

Code preferences

Code preferences are global: they apply to every SAST scan, however it is started (the `scancode` command, SAST within DAST, the Web UI, the REST API, the Lua API, the PowerShell module or the classic interface).

Preference	Type	Description
`code.checks.inflt`	boolean	**true** (default): Analyze input filtering in application source code. This is crucial for eliminating false positives, as it ensures proper handling of user input, reducing the likelihood of incorrect detections. **false**: Disables analysis of input filtering.
`code.options.target.tfsver`	string	TFS Version Compatibility. Accepts: latest, 2017, 2015, 2013, 2012, 2010. Default: **latest**., which means 2018 or higher.

Vulnerability ignore list

Each SAST vulnerability entry in a report ends with its ignore ID, the simplest and recommended way to suppress a finding; for broader scenarios, use ignore rules. For SAST, rather than `code.lists.ignore.vulnsbyrules`, the recommended practice is a `.vulnignore` file in the scanned repository.

Preference	Type	Description
`code.lists.ignore.vulnsbyrules`	stringlist	Vulnerabilities to ignore (by rule or id). Default: empty list.

Time limit

Preference	Type	Description
`syhunt.code.options.timelimit.enabled`	boolean	**false** (default): Do not use any time limit. **true**: Enable time limit. This is a master switch
`syhunt.code.options.timelimit.value`	integer	The scan time limit parameter determines the maximum duration for a scan. Default: empty., indicating no limit. Accepted time limits include formats like: **1d** for one day, **3h** for three hours, **2h30m** for two hours and thirty minutes, or **50m** for fifty minutes. Once this maximum duration is reached, the scan automatically cancels itself

Page 163 of 175

Forensic preferences

Forensic preferences are global: they apply to every FAST scan, whether started with the `scanlog` command or the classic interface.

Preference	Type	Description
`insight.lists.ignorelist.paths`	stringlist	Path Ignore List (for example, /robots.txt,robots2.txt). Default: empty list.
`insight.lists.ignorelist.srcips`	stringlist	IP Ignore List (for example, 1.1.1.1,2.2.2.2). Default: empty list.
`insight.options.logformat`	string	Log Format. Accepts: auto, common, apache, iis, ncsa, nginx. Default: **auto**., which indicates it must auto detect the log format.
`insight.options.profileatk`	boolean	**true** (default): Build attacker profile. **false**: Disables attacker profiling.
`insight.options.resolveip`	boolean	**true**: Resolve attacker IP addresses (slower). **false** (default): Disables IP resolving (faster).

Other Hybrid preferences

These global preferences apply to every scan, however it is started (the `scancode` or `scanurl` commands, the Web UI, the REST API, the Lua API, the PowerShell module or the classic interface).

Preference	Type	Description
`hybrid.report.defaultlanguage`	string	Report Language ID. See the list of available report languages. Default: **EN**., for English language reports.
`hybrid.report.company.logo.urldark`	string	Organization Logo URL for dark background (for example, `http://www.mycompany.com/logo.png`). Default: empty.
`hybrid.lists.riskfine`	stringlist	Risk Redefinition. Each line must use the format [unique check ID=severity] (for example, C-1603660271-7241=low). Default: empty list.
`syhunt.hybrid.advanced.ai.openai.apikey.encrypted`	string	OpenAI Key for enabling Syhunt's AI-powered capabilities. Default: empty.
`syhunt.hybrid.advanced.ai.options.exrepsolut`	boolean	**true** (default): Share excerpts of vulnerable code to get patch examples. Requires `ai_reviewconsent: true` in a `.syhunt-ci.yml` file at the root of each target repository. **false**: Disables share excerpts of vulnerable code to get patch examples, even if patch consent files are available.

Tracker preferences

Tracker preferences are set with `scancore -tracker:set`, using `-to` to name the tracker and `-v` for the value (examples). They apply only to that tracker.

For examples of creating trackers and setting their preferences, see Command Line: using issue trackers.

Which keys apply depends on the tracker type: for example, an email tracker uses `smtp.targetport`, while a GitHub tracker ignores it. The sections after the table list the keys each tracker type usually needs.

Preference	Type	Description
`project.name`	string	Project Name (for example, owner/repo). Default: empty.

Preference	Type	Description
`auth.username`	string	Username. Default: empty.
`auth.password`	string	Password. Default: empty.
`auth.token.encrypted`	string	Authentication Token. This can be used to set a PAT (Personal Access Token) for a service. Default: empty.
`auth.targethost`	string	Hostname. Default: empty.
`api.url`	string	API/Server URL. Default: empty.
`custom.fields`	stringlist	Custom Fields. Default: empty list.
`custom.labels`	stringlist	Custom Labels. Default: empty list.
`scan.minseverity`	string	Minimal Severity. Accepted values are: Info, Low, Medium or High. Default: **Info**.
`notifyonfailonly`	boolean	**true**: Notify only when a fail condition is met. **false** (default): Always notify (not tied to any pre-configured pass/fail conditions).

Keys by tracker type

Tracker	Keys to set
Email	`auth.targethost` (SMTP host), `auth.username` and `auth.password` (credentials), plus the email keys below.
GitHub	`project.name` (`owner/repo`), `auth.token.encrypted` (personal access token), optionally `custom.labels` (for example, `Vulnerability`).
GitLab	`project.name` (`owner/repo`), `api.url` (GitLab server URL), `auth.token.encrypted` (personal access token), optionally `custom.labels` .
Jira	`project.name` (Jira project), `api.url` (Jira server URL), `auth.username` and `auth.password` , optionally `custom.labels` , `custom.fields` and `jira.defitname` .
DefectDojo	`api.url` (server URL), `auth.token.encrypted` (API token), optionally `scan.minseverity` , `custom.labels` (tags, for example `scan`) and the DefectDojo keys below.
Faraday	`api.url` (server URL), `project.name` (workspace; defaults to `my_workspace`), `auth.username` and `auth.password` .

Email

Preference	Type	Description
`smtp.targetport`	integer	SMTP port. Default: **587**.
`message.from`	string	Sender address. Default: empty.
`message.tolist`	string	Recipient address, or a comma-separated list of addresses. Default: empty.
`message.subject`	string	Message subject. Default: empty.

Jira

Preference	Type	Description
`jira.defitname`	string	Default issue type. Default: **Task**.

Use `custom.fields` to reference custom fields by name rather than by ID. For a SelectList custom field, declare it as `customfield_10100.value=ACustomValue` .

Syhunt Hybrid 7.2

DefectDojo

Preference	Type	Description
`scan.testtitle`	string	Context: test title. Default: empty.
`scan.engagementname`	string	Context: engagement name. Default: empty.
`scan.producttypename`	string	Context: product type name. Default: empty.
`scan.productname`	string	Context: product name. Default: empty.
`scan.active`	boolean	**true** (default): mark findings as active. **false**: mark them as inactive.
`scan.verified`	boolean	**true** (default): mark findings as verified. **false**: mark them as unverified.
`scan.skipduplicates`	boolean	**true**: skip duplicates. **false** (default): don't skip duplicates.
`scan.closeoldfindings`	boolean	**true**: close old findings. **false** (default): don't close old findings.
`scan.pushtojira`	boolean	**true**: push findings to Jira. **false** (default): don't push to Jira.

Page 166 of 175

Syhunt Hybrid 7.2

Vulnerability Checks

> **Applies to:** Syhunt Hybrid 6.9

Syhunt's check database is the result of years of research and covers an extremely wide range of web application security threats. Rows in **bold** are categories; the rows that follow them are the individual checks in that category.

For mobile-specific checks, see Mobile Vulnerability Checks. To compare checks across editions, see Compare checks between editions.

Web application security checks

Check	CWE
Authentication Vulnerabilities	
Authentication Bypass / Broken Authentication through Dynamic Analysis	CWE-287
Authentication Bypass / Broken Authentication through Source Code Analysis	CWE-287
Automated Authentication Brute Force (Form & HTTP-Based)	
Password Disclosure	CWE-311
Unencrypted Login	CWE-319
Insecure Salting through Source Code Analysis	
Broken Cryptography	
Insecure Hashing Algorithms through Source Code Analysis	
Insecure Cryptographic Algorithms through Source Code Analysis	
Insecure Randomness through Source Code Analysis	
Weak Protocols through Source Code Analysis	
Weak Password Hashing through Source Code Analysis	
Breach Confirmation	
Breach Confirmation through Dynamic Analysis	
Breach Confirmation through Source Code Analysis	
Breach Confirmation through Web Server Log Analysis	
Hidden Debug Parameter Discovery through Dynamic Analysis	
Hidden Debug Parameter Discovery through Source Code Analysis	
Hidden Debug Parameter Discovery and Injection through Hybrid Analysis	
Inappropriate Content Detection	
Malicious Content Detection	
Web-Based Backdoor Detection through Dynamic Analysis	
Web-Based Backdoor Detection through Source Code Analysis	
Command Execution	CWE-78
Command Execution through Dynamic Analysis	

Page 167 of 175

Syhunt Hybrid 7.2

Check	CWE
Command Execution through Source Code Analysis	
Cross-Site Scripting (XSS)	CWE-79
Cross-Site Scripting (XSS) through Dynamic Analysis	
Cross-Site Scripting (XSS) through Source Code Analysis	
Weak XSS Filter/Validation Bypass through Dynamic Analysis	
Weak XSS Filter/Validation Bypass through Source Code Analysis	
DOM-Based XSS through Source Code Analysis	
HTML5 Specific XSS	
Cross-Site Request Forgery (XSRF) through Source Code Analysis	
File Inclusion	CWE-98
Local File Inclusion (LFI) through Dynamic Analysis	
Remote File Inclusion (RFI) through Dynamic Analysis	
Local File Inclusion (LFI) through Source Code Analysis	
Remote File Inclusion (RFI) through Source Code Analysis	
File Manipulation through Source Code Analysis	
NoSQL Injection	
NoSQL Injection through Dynamic Analysis (Error-Based)	
NoSQL Injection through Dynamic Analysis (Time-Based)	
NoSQL Injection through Source Code Analysis	
NoSQL Injection in MongoDB	
SQL Injection	CWE-89
SQL Injection through Dynamic Analysis (Error-Based)	
SQL Injection through Dynamic Analysis (Blind)	
SQL Injection through Dynamic Analysis (Time-Based)	
SQL Injection through Source Code Analysis	
SQL Injection through Source Code Analysis (HQL)	
Code Injection	CWE-94
Code Injection through Dynamic Analysis (Print-Based)	
Code Injection through Dynamic Analysis (Time-Based)	
Code Injection through Source Code Analysis	
Code Injection in ASP Classic	
Code Injection in ASP.NET	
Code Injection in Java/JSP	
Code Injection in Lua (Nginx, Apache, CGI-Lua, etc)	
Code Injection in Perl	
Code Injection in PHP	

Page 168 of 175

Syhunt Hybrid 7.2

Check	CWE
Code Injection in Python	
Code Injection in Ruby	
Code Injection in Server-Side JavaScript	
Source Code Disclosure	CWE-540
Source Code Disclosure through Injection	
Source Code Disclosure through Content Analysis	
Source Code Disclosure (ASP Classic)	
Source Code Disclosure (ASP.NET)	
Source Code Disclosure (Java/JSP)	
Source Code Disclosure (Lua)	
Source Code Disclosure (Perl)	
Source Code Disclosure (PHP)	
Source Code Disclosure (SSI)	
Extension Checking	
Double Extension Checking	
Common Backup Extensions	
Structure Brute Force	
Admin Pages	
Common Backup Files	
Common Backup Folders	
Database Disclosure	
Old/Backup Files	CWE-530
Common Form Weaknesses	
Email Form Hijacking	
Hidden Price Form Field	
AutoComplete Enabled (in sensitive form inputs)	
Unencrypted Credit Card Transaction	
Denial-of-Service (DoS)	CWE-730
Client-Side Denial-of-Service	
Denial-of-Service through Injection	
Denial-of-Service through Source Code Analysis	
Buffer Overflow	CWE-120
Multiple Disclosure Vulnerabilities	
Common Exposures	
Hardcoded Sensitive Information through Source Code Analysis	
Logging of Sensitive Information through Source Code Analysis	

Page 169 of 175

Syhunt Hybrid 7.2

Check	CWE
Local Storage Usage through Source Code Analysis	
Sensitive Data Stored in Local Storage through Source Code Analysis	
Sensitive Information Client-Side through Source Code Analysis	
Information Disclosure through Injection	CWE-200
Information Disclosure through Content Analysis	CWE-200
Information Disclosure through Source Code Analysis	CWE-497
Internal IP Address Disclosure	CWE-200
Path Disclosure through Injection	CWE-211
Path Disclosure through Content Analysis	CWE-211
Directory Listing	CWE-548
Web Technology Disclosures	
Suspicious HTML Comments	
Log Forging	CWE-117
Log Forging through Source Code Analysis	
XPath Injection	CWE-91
XPath Injection through Dynamic Analysis	
XPath Injection through Source Code Analysis	
LDAP Injection	CWE-90
LDAP Injection through Dynamic Analysis	
LDAP Injection through Source Code Analysis	
Unvalidated Redirects	CWE-601
Unvalidated Redirects through Dynamic Analysis	
Unvalidated Redirects through Source Code Analysis	
CRLF Header Injection	CWE-93
CRLF Header Injection through Dynamic Analysis	
CRLF Header Injection through Source Code Analysis	
Expression Language (EL) Injection	CWE-917
Cookie Manipulation	
Cross Frame Scripting	CWE-352
Dangerous Methods	CWE-749
Default Content	CWE-276
Directory Traversal	CWE-22
HTTP Header Injection through Source Code Analysis	
Server-Specific Vulnerabilities in IIS, iPlanet & Others	
Server-Side Request Forgery (SSRF) through Dynamic Analysis	CWE-918
Server-Side Request Forgery (SSRF) through Source Code Analysis	CWE-918

Page 170 of 175

Syhunt Hybrid 7.2

Check	CWE
Server-Side Includes (SSI) Injection	
Client-Side Request Forgery (CSRF) through Source Code Analysis	
XML Injection through Dynamic Analysis	CWE-661
XML Injection through Source Code Analysis	CWE-661
XML External Entity (XXE) Injection	CWE-827
XML External Entity (XXE) Injection through Dynamic Analysis	
XML External Entity (XXE) Injection through Source Code Analysis	
Security Misconfiguration through Source Code Analysis	
Known Vulnerable Apps	
Outdated Vulnerable Scripts through Source Code Analysis	
Known Vulnerable Apps (Apache Struts)	
Known Vulnerable Apps (ASP Classic)	
Known Vulnerable Apps (ASP.Net)	
Known Vulnerable Apps (ColdFusion)	
Known Vulnerable Apps (Dynamic HTML)	
Known Vulnerable Apps (Flash)	
Known Vulnerable Apps (Java / JSP)	
Known Vulnerable Apps (Perl)	
Known Vulnerable Apps (Python)	
Known Vulnerable Apps (Ruby)	
Known Vulnerable Apps (SSI)	
Known Vulnerable Apps (IIS)	

Supported languages

- Dynamic analysis (DAST): see Dynamic Scanning.
- Source code analysis (SAST): see Source Code Scanning.
- Mobile (MAST): see Source Code Scanning.

Supported databases (SQL injection detection)

- Access
- DB2
- dbx
- Firebird/InterBase
- FrontBase
- Informix
- Ingres
- MariaDB / MySQL
- MaxDB
- mSQL
- Oracle

Page 171 of 175

Syhunt Hybrid 7.2

- Ovrimos
- PostgreSQL
- SQL Server
- SQLite
- Swish
- Sybase

Page 172 of 175

Syhunt Hybrid 7.2

Support

Page 173 of 175

Syhunt Hybrid 7.2

Reporting Bugs and Getting Support

> **Applies to:** Syhunt Hybrid 6.8.20

Contents

- Long DAST scans
- Reporting a false positive
- Reporting a false negative
- Reporting a crawling issue
- Reporting other bugs
- Creating a debug log
- Opening a support ticket

Long DAST scans

Scanning a website with a large amount of dynamically generated content can take a long time. See Performance Tuning to understand why and how it can be addressed.

Reporting a false positive

A *false positive* is a finding that, after manual review, turns out not to be a vulnerability or weakness. To report one so it can be fixed:

1. Describe each false positive in as much detail as possible.
2. Create a debug log.
3. Open a support ticket and attach the debug files and your description.

Reporting a false negative

A *false negative* is a vulnerability that exists in the target website or codebase but wasn't detected. To report one so it can be fixed:

1. Describe the vulnerability type (for example, SQL injection).
2. **Code scans:** include the **relevant fragments** of source code that contain the undetected vulnerability.
3. **Dynamic scans:** describe the case in as much detail as possible, including:
    - the affected URL, if any;
    - the steps you used to confirm the vulnerability;
    - for injection flaws, the affected parameter, the injected data and a sample of the application's response.
4. Create a debug log.
5. Open a support ticket and attach the debug files, your description and any code fragments.

Reporting a crawling issue

1. Tell us whether the target URL is behind a barrier such as login form authentication. This helps us review the procedure, configuration or method used and suggest or make adjustments to the tool.
2. If the URL that isn't crawled properly is reachable, send it to us with test credentials (if needed), so we can test it directly from our lab.
3. If that isn't possible, create a debug log.
4. Open a support ticket and attach the debug files and a description of the issue.

Page 174 of 175

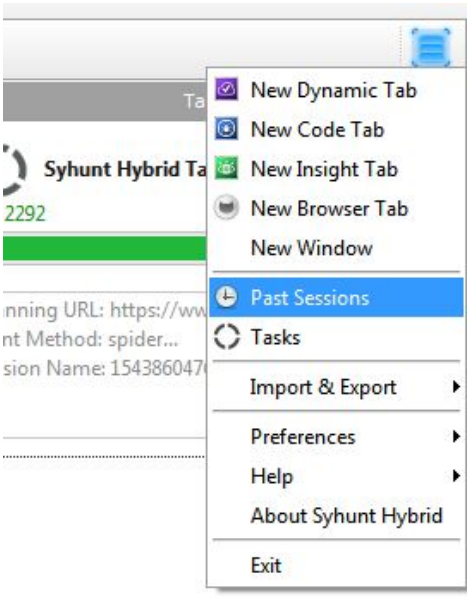
Syhunt Hybrid 7.2

Reporting other bugs

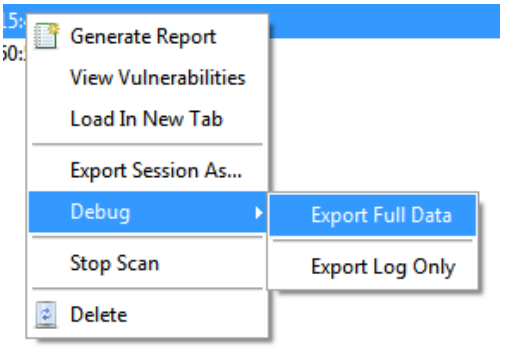
1. Describe the problem in as much detail as possible, so we can investigate and reproduce it in our lab.
2. Take a screenshot of the problem.
3. Create a debug log.
4. Open a support ticket and attach the debug files, your description and the screenshot.

Creating a debug log

1. Open the sessions screen from **Menu** → **Past Sessions**.



2. Right-click the session you want to debug and choose **Debug** → **Export Full Data** to save the file.



This creates a debug file with the `.sse` extension. Send it to Syhunt for analysis.

Opening a support ticket

A support ticket is the best way to report a bug or ask for help.

1. Go to the support ticket area and click **Submit Ticket**.
2. If you don't have an account yet, click **Register**, enter a valid email address and your name, and click **Register**. Check your email for your login details, then go back to the login page, sign in and click **Submit Ticket**.
3. Fill in the form: write your message, attach any files and click **Send**.

A Syhunt support agent will review the issue, and you'll receive a copy of every reply by email. Expect a reply within 1–2 business days. You can follow the status of your tickets under My Tickets and reply by email or from the tickets area.

Page 175 of 175