



Artemis Security Scanner

Krzysztof Zajac

cert.pl

Miscellaneous

Let's schedule one 10 minutes break.

Don't hesitate to ask questions - there are no stupid ones!

You don't have to take notes - I'll send you the slides.

Google Doc where I'll paste commands etc.:

<https://tinyurl.com/artemis-training>

The purpose of this training

1. Show the tool

2. Show the approach

I'll describe the non-technical environment as well

3. Encourage you to start similar projects

All of these are equally important!

Short description of Artemis














Purpose

After an incident, let's make sure it won't occur in other entities.

Example:

- exposed .git on an university website caused API key leak and unauthorized data access
- let's check whether other entities have exposed .git folders!

Index of /.git

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 FETCH_HEAD	2019-04-03 12:19	3.8K	
 HEAD	2019-04-03 15:04	23	
 ORIG_HEAD	2019-04-03 12:18	41	
 config	2019-04-03 12:18	312	
 description	2019-04-03 12:19	73	
 hooks/	2019-04-03 12:19	-	
 index	2019-04-03 15:04	226K	
 info/	2019-04-03 12:19	-	
 logs/	2019-04-03 12:19	-	
 objects/	2019-04-03 12:19	-	
 packed-refs	2019-04-03 12:19	22K	
 refs/	2019-04-03 12:19	-	

A domain →  **artemis** →

1. The following addresses contain version control system data:

- [https://\[REDACTED\]:443/.git/](https://[REDACTED]:443/.git/)

(...)



1. The following addresses contain version control system data:

- [https://\[REDACTED\]:443/.git/](https://[REDACTED]:443/.git/)

(...)

2. The following addresses contain old Joomla versions:

- [https://\[REDACTED\]:443](https://[REDACTED]:443) - Joomla 2.5.4

(...)

.gov.pl,
schools,
hospitals,
universities,
banks,
...



artemis



1. The following addresses contain version control system data:

- [https://\[REDACTED\]:443/.git/](https://[REDACTED]:443/.git/)

(...)

2. The following addresses contain old Joomla versions:

- [https://\[REDACTED\]:443](https://[REDACTED]:443) - Joomla 2.5.4

(...)

What do we check?

A couple dozen modules

- Subdomain enumeration (from various data sources)
- Domain expiration check
- Bad DNS configuration check:
 - Zone transfer
 - Subdomain takeover
- SPF/DMARC
- Bad/expired TLS certificates, https:// redirect

A couple dozen modules

- Port scanning
- WordPress, WordPress plugin, Drupal, and Joomla version check
- Closed WordPress plugins
- **Nuclei support:** thousands of vulnerabilities and misconfigurations (from Open Redirects to RCEs)
- SQLi and XSS

A couple dozen modules

- Scripts loaded from nonexistent domains
- Directory index
- Weak passwords
- Exposed Git/SVN repositories
- Exposed login panels (RDP, phpMyAdmin, ...)
- Accidentally published files (eg. SQL dumps, backups or wp-config.php.bak)

- ...

Didn't find something on the list?

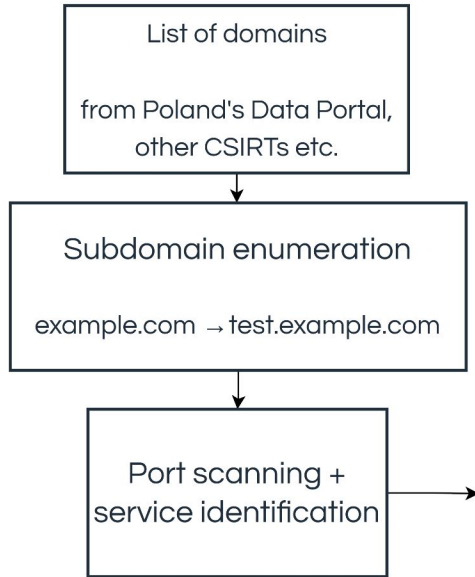
Contact us or **submit a pull request!**

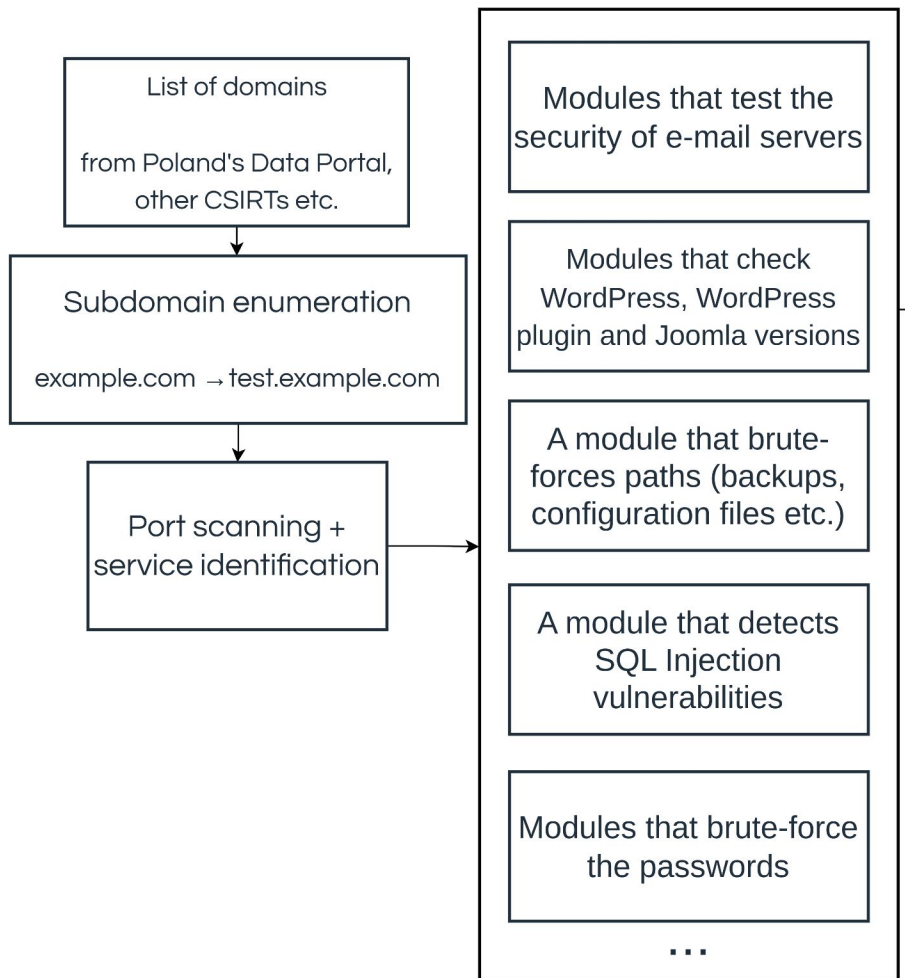
It's possible to integrate **any other tool** (commercial or open-source), and Artemis has an example how to do that.

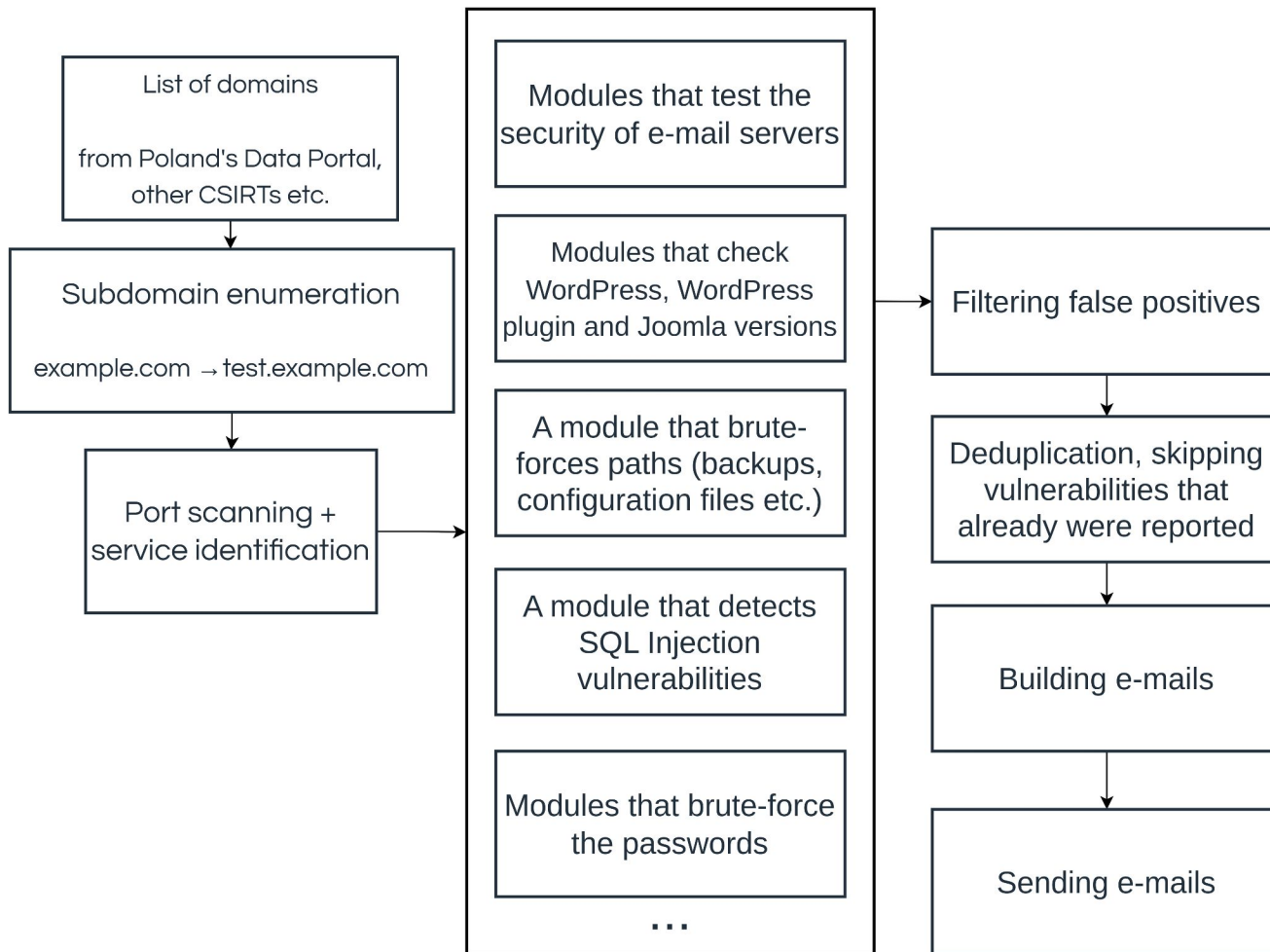
Where the list comes from?

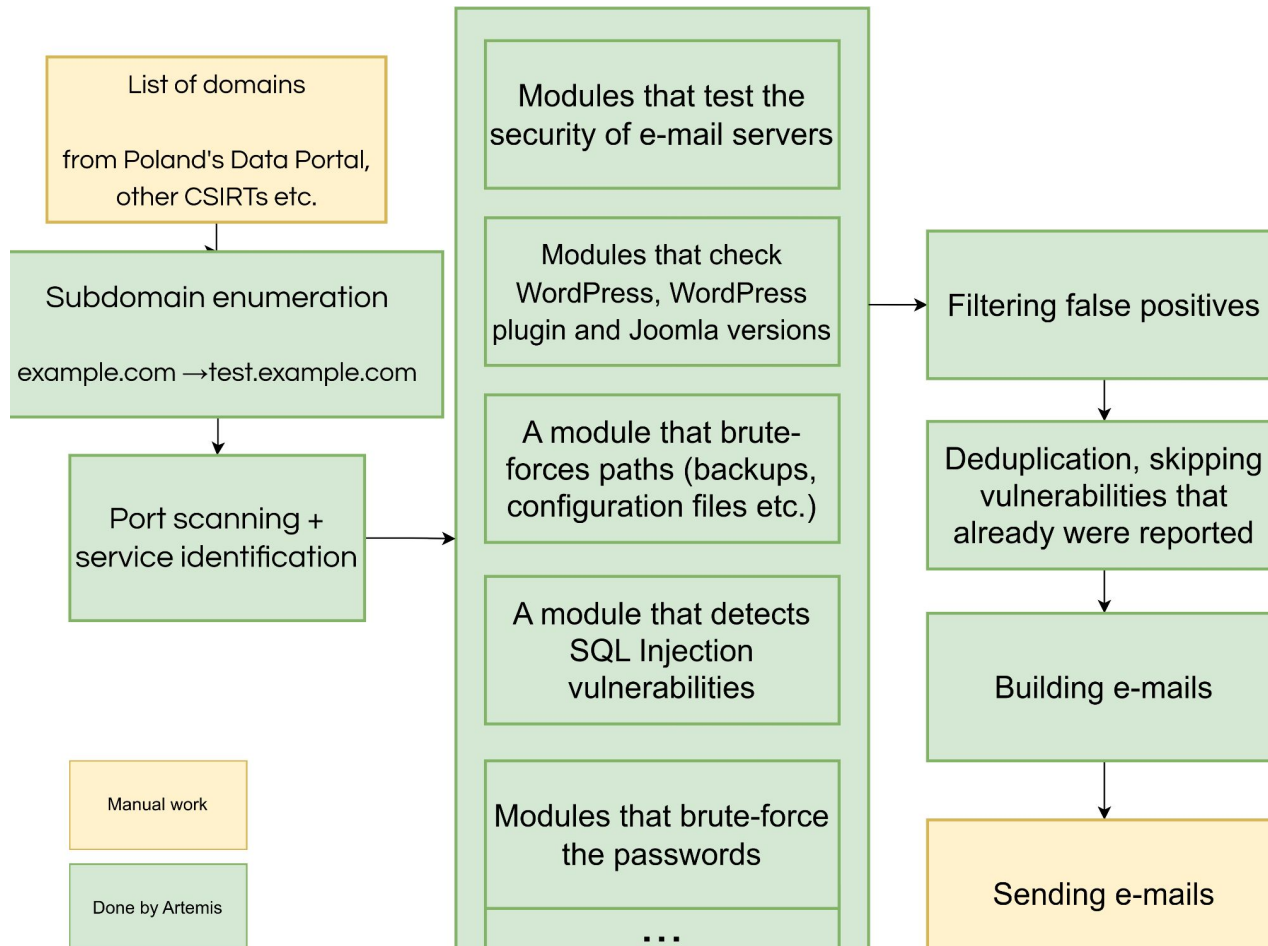
Our experiences in handling incidents.

E.g.: someone got hacked because of a SQL Injection? Let's improve SQL Injection detection capabilities!









Example report

The following addresses contain version control system data:

- [https://\[REDACTED\]:443/.git/](https://[REDACTED]:443/.git/)

Making a code repository public may allow an attacker to learn the inner workings of a system, and if it contains passwords or API keys - also gain unauthorized access. Such data shouldn't be publicly available.

Example e-mail

Such reports are sent by CERT PL to scanned entities.

Any questions so far?

Statistics

Since January 2023 we reported ~**448.5k** vulnerabilities and misconfigurations, including:

- ~**27.6k** high-severity,
 - ~**284.9k** medium-severity,
 - ~**136k** low-severity.
- For example we have over **1000** confirmed SQL Injections (where we managed to **dump data from the database**).

Communication

- We already sent over **110k e-mails**.
- If an entity doesn't fix a serious issue, **we call them**. We already made around **6k** such calls.
- Reactions are mostly positive (but we sometimes receive bug reports).
- Important: sometimes our e-mail **gives “political” support to the admins** even if they know about a problem.

Let's start similar scanning in
your constituency!

A machine with Linux

Who has one? The next slides will describe Artemis setup.

If you don't, you will be able to **use one I set up.**

Installing Git, Docker and Docker Compose

(the latter two from docker.com, not
system repository)

Artemis on your machine

[https://artemis-scanner.readthedocs.io/
en/latest/quick-start.html](https://artemis-scanner.readthedocs.io/en/latest/quick-start.html)

Walkthrough!

Artemis on your machine

Install **Artemis-modules-extra** as well.

(that gives you **SQL injection**, **SSL** and **subdomain takeover** check capabilities).

Add the following to .env (why?)

```
CUSTOM_USER_AGENT=Mozilla/5.0 (compatible; Artemis;  
YOUR-CSIRT-NAME; +your-email@your-csirt.tld)
```


Example Artemis instance

<https://scanner.artemis-trainings.cert.pl/>

Username: **artemis** password: **[redacted]**

The provided machine scans **only port 80 and 443** with **fewer modules** (only these doing standard HTTP GET or DNS requests).

What modules are disabled?



DigitalOcean Abuse <abuse... Wed, May 15, 10:51AM



to me ▾

Hello,

Greetings from the **DigitalOcean** team.

Thank you so much for your patience, as we investigated this incident and reviewed the information you have shared with us so far, we have come to the conclusion that your deployment and actions were in violation of our Terms of Service Agreement [1] and Acceptable Use Policy [2] and as a result, we are unable to remove the lock from your account.

I sincerely apologize for any inconvenience caused by this.

Additionally, please note that we're unable to share details regarding what factors we considered that led to this decision or how we came to this conclusion as this is critical to maintaining the integrity of our platform security operations.

Let's scan some domains!

List of domains

Who has one?

What kind of domains and how many?

Example: data.gov

- <https://catalog.data.gov/dataset/common-core-of-data-ccd-district-nonfiscal-data-files-and-documentation-2018-19>
- Example in other countries:
<https://dane.gov.pl/en>

Other data sources

- Subdomain enumeration (e.g. crt.sh: <https://crt.sh/?q=%25.gov.pl>),
- Custom databases (example: rspo.gov.pl for schools),
- Be creative (example: mamprawowiedziec.pl),
- Contacting the entities (slow)
- ...

Inspirations

- All gov.[your-tld] domains
- Local government entities
- Municipal corporations: water management, waste collection, ...
- Key Service Operators
- Banks
- Universities, schools, preschools and other educational entities

Inspirations

- Hospitals
- Local and country-level newspapers, TVs, information portals etc.
- Websites of politicians, political parties, candidates etc.
- Professional self-governments (e.g. medical chambers)

Legal basis

Various legal bases for various scans:

- Rules of the gov.pl domain,
- Act of National Cybersecurity System,
- Agreements with other CSIRTs or ministries,
- Etc.

Don't shoot yourself in the foot

Don't design law that:

- Allows scanning of only a small subset of entities (e.g. *important* ones)
- Requires actions that are **not viable** in case of broad scans, such as signed agreements with entities.

Possibility to perform **broad scans** was **crucial** for the success of Artemis!

CERT.PL (Keep an eye on the above when implementing EU NIS2)

Any questions so far?

Rules

You may see some real vulnerabilities from outside of your constituency e.g. on the example Artemis instance you are using.

Don't share them.

Let's scan

Scan your list.

If you don't have a list: try **four universities from your country.**

Why universities?

Selecting which modules to run

Any questions so far?

Viewing the task queue

Viewing the raw results

Let's export the reports



artemis →



Let's export the reports

Export: **the most important feature of Artemis!**

And what with the problems that didn't make it to the reports?

Artemis checks whether a problem is interesting enough to be included in the reports.

API: let's check SPF/DMARC
(why?)

```
curl -X POST --header "X-Api-Token: 123" \  
  --header "Content-Type: application/json" \  
  --data '{"enabled_modules": ["mail_dns_scanner"],  
  "targets": ["data.gov.lt"]}' \  
127.0.0.1:5000/api/add
```

```
curl -X GET --header "X-Api-Token: 123" \  
    127.0.0.1:5000/api/num-queued-tasks  
curl -X GET --header "X-Api-Token: 123" \  
    127.0.0.1:5000/api/analyses
```

```
curl -X POST \  
  --header "X-API-Token: 123" \  
  --header "Content-Type: application/json" \  
  --data '{"language": "en_US",  
  "skip_previously_exported": false}'  
127.0.0.1:5000/api/export
```



```
curl -X GET --header "X-API-Token: 123"  
127.0.0.1:5000/api/exports
```

```
curl -L -X GET --header "X-API-Token: 123" \  
127.0.0.1:5000/api/export/download-zip/1  
--output - > export.zip
```

Any questions so far?

Translations

OK, you have a constituency speaking a language other than English?

Let's translate Artemis! docs

Translations

Lazy way:

translate the following file to French, leaving stuff in quotes after "msgid" in English but putting the translation after "msgstr":



ChatGPT

Here's the translation of the file into French:

```
#: artemis/reporting/modules/wp_scanner/template_insecure_wordpress.jinja2:3
```

```
msgid ""
```

```
"The following addresses contain WordPress versions that are no longer "
```

```
"supported and are marked as insecure in the version list from"
```

```
msgstr ""
```

```
"Les adresses suivantes contiennent des versions de WordPress qui ne sont plus "
```

```
"prises en charge et sont marquées comme non sécurisées dans la liste des "
```

```
"versions de"
```

```
#: artemis/reporting/modules/wp_scanner/template_insecure_wordpress.jinja2:19
```

```
#: artemis/reporting/modules/wp_scanner/template_old_wordpress.jinja2:14
```

```
msgid ""
```

```
"If a site is no longer used, we recommend shutting it down to eliminate "
```

```
"the risk of exploitation of known vulnerabilities in older WordPress "
```

```
"versions. Otherwise, we recommend enabling WordPress core and plugin "
```

```
"automatic updates."
```

```
msgstr ""
```

Translations

You don't have to translate everything!

Artemis will raise an error on missing translations.

Send a pull request with new translations!

Approach to time

Our configuration in production (a safe bet):

```
LOCK_SCANNED_TARGETS=True
```

```
REQUESTS_PER_SECOND=1
```

```
SCANNING_PACKETS_PER_SECOND=10
```

Artemis in production

Scaling the containers - if have RAM, don't be afraid of modifying the number of instances in `scripts/start`:

```
--scale=karton-bruter=20 # or more
```


Artemis in production

Exercise:

If you have your own Artemis instance, scale one module (e.g. bruter) to 20 instances.

That means 20 path brute-forcing tasks (e.g. /wp-config.php.bak) will be done in parallel.

Use <http://127.0.0.1:5000/queue> for information what modules to scale.

Artemis in production

Most important: **do things well enough.**

Examples:

- A module is broken? Scan using the rest until it gets fixed.
- A check crashed because of e.g. a timeout? If it's not a bug, we ignore this - we will retry during the next scan.

Approach to time

Our scanning is slow (and it's OK for us).

We send vulnerabilities e.g. every 2 weeks.

We don't wait for the scan to finish.

Artemis in production

Technical configuration:

- Scaling the modules
- Being open: Reverse DNS, User-Agent
- Monitoring (at least RAM + disk), ...

Artemis in production

Non-technical environment:

- Being open: <https://cert.pl/skanowanie/> (translation [here](#))
- Making sure the vulnerabilities get fixed - e.g. **calling the scanned entities**
- Being prepared to respond to administrator inquiries
- Allow submitting domains voluntarily (even in a **low-tech way** - e-mails)

Advanced usage: interesting features

- Stopping scanning
- Skipping already reported issues
- Automatic task result archiving
- Blocklist - blocking some issues from being reported
- Deduplication
- `NUCLEI_ADDITIONAL_TEMPLATES,`
`NUCLEI_TEMPLATES_TO_SKIP,`
`NUCLEI_SUSPICIOUS_TEMPLATES`
- ...

Other features

Stumbled upon a problem? **Contact us** at artemis@cert.pl - maybe we have already solved it?

We needed to solve **a lot of** weird problems.

False positives

If you see a pattern of false positives in the exported reports, file a bug.

We want the exported reports to be free from false positives. The exported reports don't contain everything.

False positives

Artemis will alert you if something is suspicious (e.g. a Nuclei template known from false positives).

You can blacklist issues/domains/IP ranges from being scanned/reported.

Conclusions

- Lots of low-hanging vulnerabilities
- Many good offensive tools are available
even plain Nuclei scan or WordPress/Joomla version check would find many vulnerabilities
- Iterative development contributed to the project success

Instead of building the best scanner possible, we built a MVP with a subset of modules and ran initial scans. During scans, we observed bugs, fixed them, but also added new modules.

Good luck!

Links

<https://github.com/CERT-Polska/Artemis>

<https://github.com/CERT-Polska/Artemis-modules-extra>

artemis@cert.pl

<https://discord.com/invite/GfUW4mZmy9>

Questions?

Don't hesitate to contact us!

Appendix: writing a new module

- Easiest way: a Nuclei template.

Exercise: Write a Nuclei template that detects websites that use WordPress.

Appendix: writing a new module

- Harder but more expressive way: an Artemis module

Artemis has an example module to copy:

<https://artemis-scanner.readthedocs.io/en/latest/user-guide/writing-a-module.html>