# Vulnerabilities in Printix Cloud Print Management

Pentagrid AG — 2022-08-18 08:42

During a penetation test of a cloud environment, Pentagrid observed vulnerabilities in the cloud printing solution developed by Printix.net ApS, a Kofax subsidiary.

## Impact

These vulnerabilities lead to an elevation of privileges to `SYSTEM` and access to other user's printed documents.

## Timeline

- 2022-05-05: Initial contact of Printix team via E-mail to info@printix.net.
- 2022-05-12: Sent reminder via Twitter to @printixnet and @Kofax.
- 2022-05-19: Notified a founder and the CEO via E-mail.
- 2022-05-20: Printix replied that the info@printix.net is not monitored for security and that the E-mail was initially deleted.
- 2022-05-20: Pentagrid recommends following RFC 9116. Out of courtesy the 14 days feedback limit is extended to 24 May EOD to provide feedback, which steps are planned and when this will happen.
- 2022-05-22: Printix replies they plan to solve the problems as soon as possible and mentions a start in the week of 2022-05-30.
- 2022-05-23: Pentagrid communicates disclosure deadline 2022-08-03 and asks for a detailed plan (fixes, release of updates, estimates on how many customers have a patched state on 2022-08-03).
- 2022-05-31: Printix replies with a fix prioritization, which changes are planned, and that a rollout is planned for July.
- 2022-08-01: Pentagrid asked for a status.
- 2022-08-02: Printix confirmed that they adressed issues according to their plan with release 1.3.1169, which is in testing stage and that Printix needs 2-3 weeks for public deployment.
- 2022-08-11: Pentagrid postponed publication until 2022-08-18.
- 2022-08-18: CVE-2022-35167 was assigned to one of the issues.
- 2022-08-18: Advisory published.

## Affected Components

These vulnerabilities have been observed in the Printix client version 1.3.1149.0 for Windows and the corresponding cloud portal at the time of discovery.

## Vulnerabilities

### 1. Printix registry keys not read/write protected (CVE-2022-35167)

```
CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H, 7.8 High
```

After installing Printix on Windows, Registry keys belonging to each user are accessible to all other users on the system and can be modified.

#### Impact

An attacker is able to read tokens of other users and can change program parameters, such as dependent libraries, external binary dependencies or the uninstallation script that gets executed when the application is removed from the system. Ultimately, tampering with the configuration of Printix in the registry can lead to an elevation of privileges.

#### Technical Details

The Printix software is running as a background service and a client application, which, on Windows systems, is reading its configuration values from registry keys at `HKLM/SOFTWARE/printix.net/Printix Client/CurrentVersion`. Every user has their own subkey within this registry key.

Every regular user can read the Printix registry keys of other users that were logged in to the same computer at least once. One of the entries within the key is an access token. Other global entries within the printix.net registry key include the paths of certain third-party libraries and binaries the application relies on. An attacker could overwrite the paths and let them point to their own, malicious libraries, which consequently might be loaded by the application.

##### 1.1 Weak registry key permissions enable privilege escalation

A clean installation of the Printix client was done to analyse the default configuration as shipped by the developers. This revealed that a privilege escalation is possible by changing the registry keys ProgramDir and DataDir.

After installation they initially both point to a write-protected location within the `Program Files` directory. As the keys are write-accessible by regular users, it is possible to change them, for example to C:\temp. After doing this, and restarting the Printix service, new configuration files are created by the service in that directory. One of the new files is called `PrintixServiceTask.xml`, which is a Microsoft Task definition, describing the

Printix service running on the machine. After changing the paths to C:\temp the file is also read from this location, and as the new directory is not write-protected, the file can also be modified by any regular user.

One part of the XML file is the command that will be run by Windows to start the service as shown in the following listing.

**Source:** C:\temp\PrintixServiceTask.xml

```
<?xml version="1.0" encoding="UTF-16"?>
<Task version="1.3" xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task">
    <RegistrationInfo>
        <Date>2017-09-21T13:43:01</Date>
        <Author>Printix</Author>
        <URI>\Printix Service</URI>
    </RegistrationInfo>
    <Triggers>
        <EventTrigger>
        <Enabled>true</Enabled>
        <Subscription>&lt;QueryList&gt;&lt;Query Id="0" Path="Application"&gt;&lt;Select Path="Application"&gt;*
[System[Provider[@Name='PrintixClient'] and EventID=249]]&lt;/Select&gt;&lt;/Query&gt;&lt;/QueryList&gt;
</Subscription>
        </EventTrigger>
    </Triggers>
    <Principals>
        <Principal id="Author">
            <UserId>S-1-5-18</UserId>
            <RunLevel>LeastPrivilege</RunLevel>
        </Principal>
    </Principals>
    <Settings>
        <MultipleInstancesPolicy>IgnoreNew</MultipleInstancesPolicy>
        <DisallowStartIfOnBatteries>false</DisallowStartIfOnBatteries>
        <StopIfGoingOnBatteries>true</StopIfGoingOnBatteries>
        <AllowHardTerminate>true</AllowHardTerminate>
        <StartWhenAvailable>false</StartWhenAvailable>
        <RunOnlyIfNetworkAvailable>false</RunOnlyIfNetworkAvailable>
        <IdleSettings>
            <StopOnIdleEnd>true</StopOnIdleEnd>
            <RestartOnIdle>false</RestartOnIdle>
        </IdleSettings>
        <AllowStartOnDemand>true</AllowStartOnDemand>
        <Enabled>true</Enabled>
        <Hidden>false</Hidden>
        <RunOnlyIfIdle>false</RunOnlyIfIdle>

<DisallowStartOnRemoteAppSession>false</DisallowStartOnRemoteAppSession>
        <UseUnifiedSchedulingEngine>true</UseUnifiedSchedulingEngine>
        <WakeToRun>false</WakeToRun>
        <ExecutionTimeLimit>PT1H</ExecutionTimeLimit>
        <Priority>1</Priority>
    </Settings>
    <Actions Context="Author">
        <Exec>
            <Command>cmd</Command>
            <Arguments>/c "netsh start PrintixService"</Arguments>
        </Exec>
    </Actions>
</Task>
```

The command will be executed by the Printix Windows service, which runs as `SYSTEM` user. For testing purposes, the command section of the XML file was changed to only write the name of the user running the command itself into a text file located at C:\temp\output.txt.

```
<Exec>
    <Command>c:\windows\system32\cmd</Command>
    <Arguments>/c "whoami > c:\temp\output.txt"</Arguments>
</Exec>
```

After restarting the Printix service another time, and stopping it afterwards to make sure any open file handles are closed and flushed, the new file appears as expected, with the following contents.

**Source:** C:\temp\output.txt

```
nt authority\system
```

This confirmes that the command listed in the XML file is executed as `SYSTEM` user. As a result, a regular user is able to run commands with higher then intended privileges. The only problem an attacker has, is that a regular user is not able to restart a service at will. However, if this attack is performed on a physical machine, it would be possible to achieve the service restart by restarting Windows or power cycling the whole machine.

### 1.2 Abusing the uninstall related registry keys

Another problem appears in the registry keys responsible for uninstalling parts of the Printix application. One of the keys is present at `HKLM/SOFTWARE/Microsoft/Windows/CurrentVersion/Uninstall/printix`. This key is not write-protected, allowing any user to add new entries to it, and changing the behaviour of the uninstallation procedure of Printix. If an attacker, for example, switches the value of the existing entry `SystemComponent` from 1 to 0 and adds a new entry called `DisplayName`, a new item would appear in the Programs and Features section of the Windows Control Panel application.

If the attacker further modifies the registry and changes the existing entry `UninstallString` to the path of a binary under their control, the malicious binary would get executed if an administrator decides to uninstall the software.

## Precondition

An attacker has access to the system, modifies the mentioned registry keys and is able to inject their own malicious binaries, that are written in a way that the anti-malware solution in use is not able to detect them.

# 2. Opened ports by Printix

```
CVSS:3.1, 0.0 Information
```

The default installation of Printix on Windows configures the Windows Firewall to open multiple ports, but does not limit the rules to a specific application.

## Impact

An attacker could utilise the whitelisted outbound and inbound ports for establishing new connections to servers under their control and potentially use it for data exfiltration.

## Technical details

The printix.net application is installed in C:\Program `Files\printix.net\Printix Client` and contains a batch script called `open_firewall.cmd` that configures the Windows Firewall to allow communication to and from the Printix Client to services outside the local system.

**Source**: C:\Program `Files\printix.net\Printix Client\open_firewall.cmd`

```
@ECHO ************************************************
@ECHO * Delete Printix Client inbound ports
@ECHO ************************************************
netsh advfirewall firewall delete rule name="Printix SNMP, UDP"
netsh advfirewall firewall delete rule name="Printix PDP, UDP"
netsh advfirewall firewall delete rule name="Printix Raw Print , TCP"
netsh advfirewall firewall delete rule name="Printix Jobforward, TCP"
netsh advfirewall firewall delete rule name="Printix Redirector, TCP"
netsh advfirewall firewall delete rule name="Printix Task Manager, TCP"
netsh advfirewall firewall delete rule name="Printix UI Communication, TCP"
netsh advfirewall firewall delete rule name="Printix IPP Print, TCP"

@ECHO ************************************************
@ECHO * Adding Printix Client inbound ports
@ECHO ************************************************
netsh advfirewall firewall add rule name="Printix PDP, UDP" protocol=UDP dir=in localport=21337 action=allow
netsh advfirewall firewall add rule name="Printix Jobforward, TCP" protocol=TCP dir=in localport=21335 action=allow
netsh advfirewall firewall add rule name="Printix Redirector, TCP" protocol=TCP dir=in localport=21336 action=allow
netsh advfirewall firewall add rule name="Printix UI Communication, TCP" protocol=TCP dir=in localport=21338
action=allow
netsh advfirewall firewall add rule name="Printix IPP Print, TCP" protocol=TCP dir=in localport=21339 action=allow

@ECHO ************************************************
@ECHO * Adding Printix Client outbound ports
@ECHO ************************************************
netsh advfirewall firewall add rule name="Printix SNMP, UDP" protocol=UDP dir=out localport=161 action=allow
netsh advfirewall firewall add rule name="Printix PDP, UDP" protocol=UDP dir=out localport=21337 action=allow
netsh advfirewall firewall add rule name="Printix Raw Print , TCP" protocol=TCP dir=out localport=9100 action=allow
netsh advfirewall firewall add rule name="Printix Jobforward, TCP" protocol=TCP dir=out localport=21335 action=allow
netsh advfirewall firewall add rule name="Printix Redirector, TCP" protocol=TCP dir=out localport=21336 action=allow
netsh advfirewall firewall add rule name="Printix UI Communication, TCP" protocol=TCP dir=out localport=21338
action=allow
netsh advfirewall firewall add rule name="Printix IPP Print, TCP" protocol=TCP dir=out localport=21339 action=allow
```

The script seems to have been run during the standard installation on the tested device as the configured rules can also be seen when checking Windows Firewall settings.

### Precondition

An attacker has access to the system and is able to run an application to communicate through one of the opened ports without being detected by an installed Antivirus application.

## 3. Incomplete JWT validation allows receiving printed documents of other users

```
CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N, 6.5 Medium
```

The login mechanism to Printix via the Printix Client is not verifying the signature of a JSON Web Token (JWT) that is used to transmit data to the server. That enables tampering of the data that is sent, such as which user is being logged in. Printix uses user-specified data to construct a Windows Registry path for storing printing-related configuration.

### Impact

A manipulated user information in the JWT leads to a manipulated Windows registry path. An attacker can use this vulnerability to modify printer configuration and to receive printed documents of other users.

### Technical details

Printix is a cloud printing solution that was installed on endpoints in a tested environment. Users could sign into Printix using their Microsoft accounts by clicking onto the system tray icon of Printix and then selecting the Sign in button.

When doing that, a browser window is opened for the URL sign-in.printix.net. Several URL parameters are added as well, with one of them being a JSON Web Token as shown below.

**Source**: Example login URL

```
https://auth.printix.net/oauth/authorize/tenant/<tenant_id>?response_type=code&client_id=PrintixClient&client_secret=
<secret>&state=<username>&redirect_uri=http
```

**Source**: Decoded JWT Content

```
{
 "tenantID": "<tenant_id>",
 "state": "user2",
 "prompt": "login",
 "client_id": "PrintixClient",
 "SPRING_SECURITY_SAVED_REQUEST": {
     "requestUrl": "https:\/\/auth.printix.net:443\/oauth\/authorize\/tenant\/<tenant_id>?
response_type=code&client_id=PrintixClient&client_secret=1234&state=user2&redirect_uri=http:\/\/localhost:21339\/oauth\

     "queryString":
"response_type=code&client_id=PrintixClient&client_secret=1234&state=user2&redirect_uri=http:\/\/localhost:21339\/oauth
 }
 }
```


Usually, all these attributes of the payload cannot be modified, as servers also check the signature that is part of the JWT. In the case of Printix however, this signature is not verified at all. The signature can be replaced with any text and the server will still accept the token as valid.

When changing the state attribute to the value of another user (user B), and replacing it with the original JWT in the URL-Parameter, but afterwards signing in to Printix with the original Microsoft credentials (user A), the sign-in is confirmed by Printix.

The Printix Service running on the system will, after signing in, replace the registry entries of the user B with the username of user A. The consequence is that whenever user B prints a document, it will appear in the Printix online queue of user A.

Further investigation with tampering of the JWT revealed that this also enables an attacker to write new registry keys with `SYSTEM` privileges to different places. When passing a path traversal, e.g. `..\\..\\foobar` (escaped back-slashed), in the state field of the JWT, the Printix service will create registry keys at that location.

### Precondition

An attacker has access to the system and the targeted user also has an account on the same machine and prints a document.

## Recommendation

Pentagrid recommends installing security fixes once updates are provided by Printix. According to Printix, issues will be fixed in release 1.3.1169. However, this version has not been tested again.

## Credits

These vulnerabilities have been found by Ole Diederich (Pentagrid).

## Hiring

To extend our technical team at Pentagrid, we are looking for talented people with an interest in new things, passion for technology and commitment. Currently, we are looking for Junior and Senior IT Security Analysts in Berlin and Buchs SG.




Contact:

**Pentagrid AG**
Bahnhofstrasse 7
CH-9470 Buchs SG
Switzerland

**Pentagrid GmbH**
Am Treptower Park 75
DE-12435 Berlin
Germany

Phone: +41 81 511 2556
E-mail: contact@pentagrid.ch