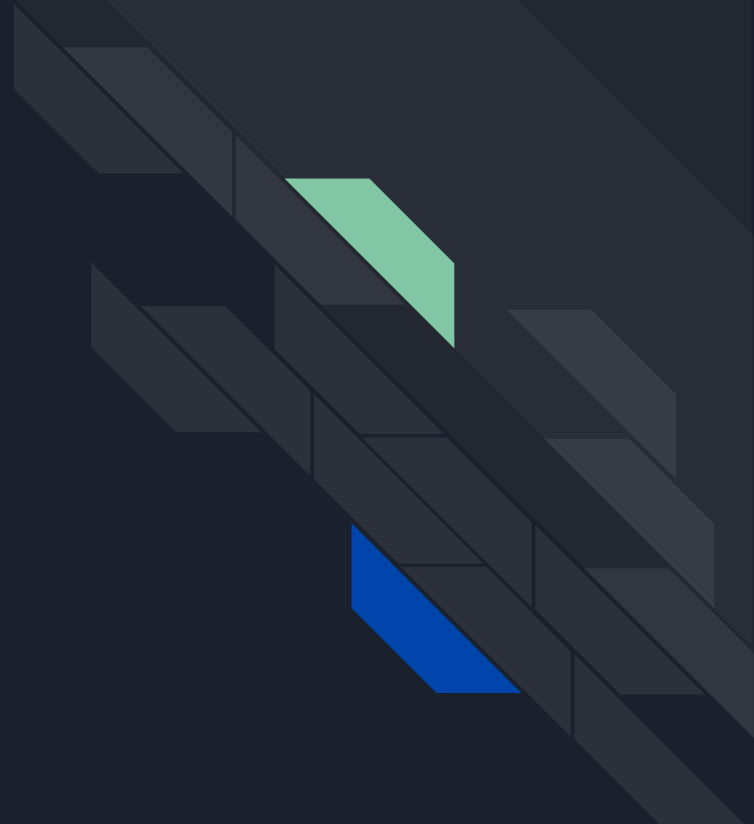




DrayTek Déjà Vu: New Tricks, Same spot, Still RCE

CVE-2025-10547

Pourquoi DrayTek ?



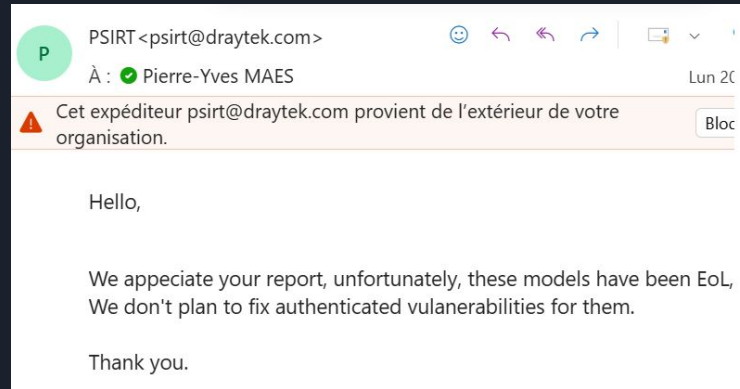
Pourquoi DrayTek ?

- Travaux précédents sur le Vigor2960 beaucoup de RCE authentifiées
- Malheureusement sur un équipement EOL

Dear Draytek,

I found two vulnerabilities affecting the device Vigor2960.
Both two vulnerabilities identified are remote code execution (RCE) issues located within the administrative interface of the device.
They are present on the **Vigor 2960** equipment and allow an authenticated attacker to execute arbitrary code on the system.
Please find attached the proof of concept as well as the write-up of the two vulnerabilities.

Kind regards,





Pourquoi DrayTek ?

- RCE découvertes très facilement
 - Peut-être que dans leurs matériels récents il existe des portions de code problématiques ?
 - Équipement VPN, peut-être moyen d'attaquer le WAN ?
- Travaux précédents
 - [HEXACON2022 - Emulate it until you make it! Pwning a DrayTek Router by Philippe Laulheret \(CVE-2022-32548\)](#)
 - [When \(Remote\) Shells Fall Into The Same Hole: Rooting DrayTek Routers Before Attackers Can \(CVE-2024-41592\)](#)



Pourquoi DrayTek ?

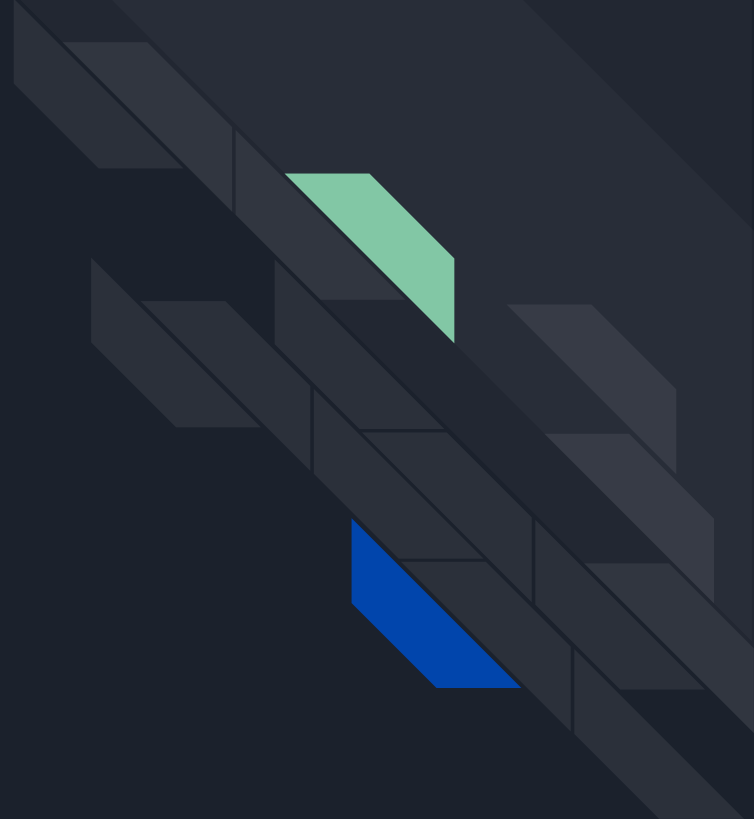
- Coeur de métier
 - Internet Service Provider pour leur box internet principalement UK
 - PME pour leurs équipements VPN
- Cible de choix pour des attaquants (source [Forescout](#))
 - ZuoRA
 - HiatusRAT
 - Volt Typhoo

Pourquoi DrayTek ?

- Une cible potentiellement intéressante :)



Analyse du firmware





Analyse du firmware

- Les nouveaux firmware sont chiffrés
- Ouverture du firmware Philippe Laulheret Hexacon 2022
 - Explique comment ouvrir le firmware
 - Parle de sources GPL présentes chez Draytek
- Choix de la cible (Vigor 2962) car l'archive GPL était la plus lourde
- Script présent dans cette archive permettant de déchiffrer le firmware

Analyse du firmware

- Sources GPL du Vigor 2962

gplsource.draytek.com/?dir=Vigor2962

DrayTek File Server

Root > Vigor2962

File name ▲	Size	Last updated
..		
Vigor2962_431_GPL_release.tar.bz2	2.25 GiB	16.01.24 10:18:15

Mobile view | Page loaded in 14.63 ms | DrayTek Corp

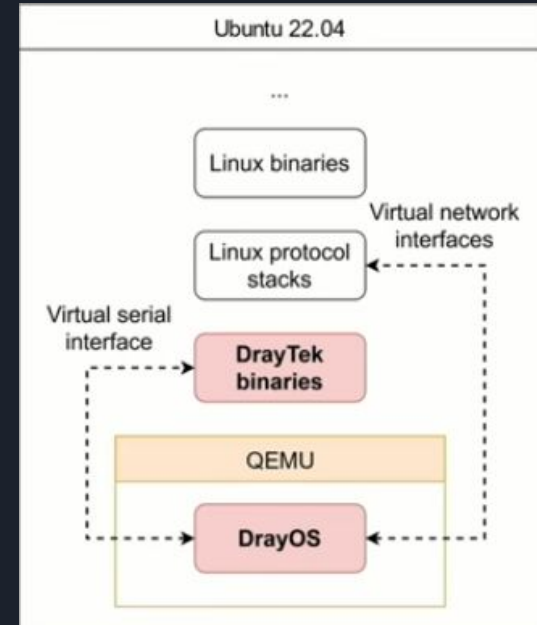
Analyse du firmware

- Dossier /draytek/drayapp contenant
 - [qemu.sh](#) le script qui lance qemu
 - soho2962.bin Real Time Operating System contenant le code du routeur

```
basketmaker@totemkid:/tmp/tmpy/full_system$ ls draytek/drayapp/  
button_detect  draycert_def.cfg  qemu_renice.sh  sqlite3  
chacha20       dray_dpdk         qemu.sh         svlogd_conf_drayos  
tcm           drayivshmem      recvCmd        syslog_server  
console       draylog_ctl      runcommand     testfunc  
dboot         ipsec-dray.cfg   scripts        updatePS  
dpdk.sh       magic_file       soho2962.bin  
basketmaker@totemkid:/tmp/tmpy/full_system$
```

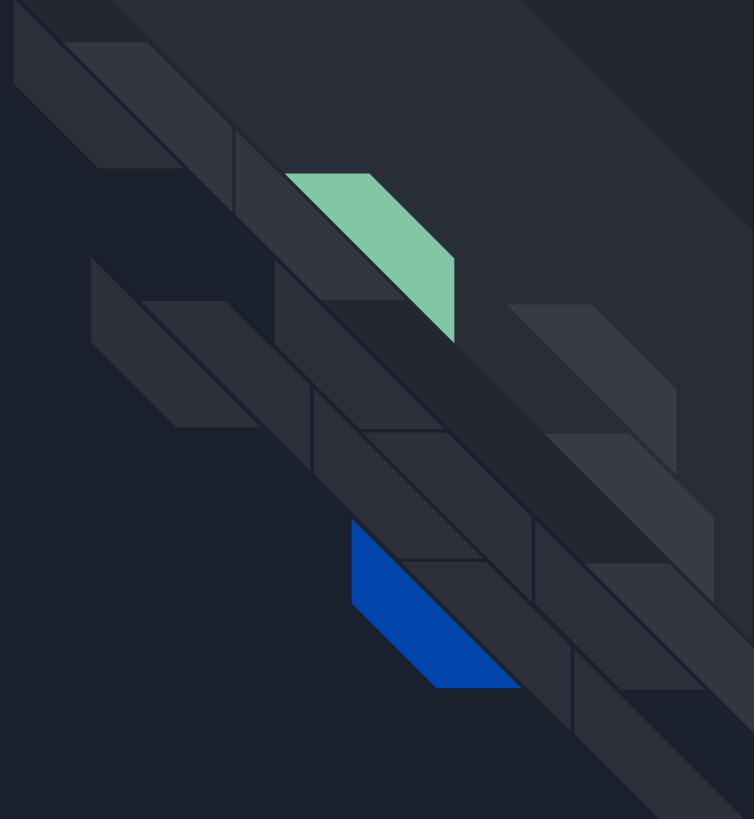
Analyse du firmware

- L'équipement contient un système Linux Ubuntu 22.04
- L'équipement lance le système DrayOs avec qemu
- Possibilité d'émuler le firmware sans avoir à l'acheter !
- Image récupérée de [Forescout](#)



CVE-2025-10547

introduction





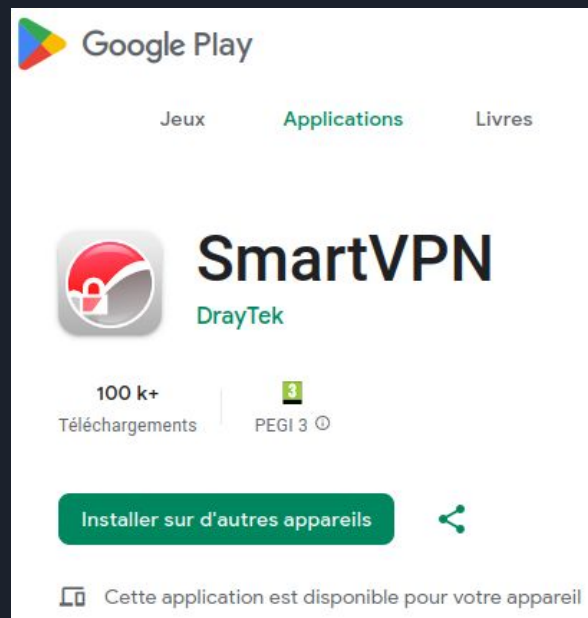
CVE-2025-10547 introduction

- Plusieurs scripts CGI accessibles sans authentification
- Une première RCE repérée rapidement mais authentifiée (nous ne voulons que du non authentifié)
- Une interface WAN qui permet aux utilisateurs de se connecter via le VPN

CVE-2025-10547 introduction

- Application mobile permettant de se connecter au matériel
- Le reverse de l'application permet de repérer un endpoint qui utilise la méthode HTTP CONNECT
- "CONNECT /" étant une utilisation étrange de cette méthode HTTP

```
pn_client/src/base$ grep -rn "CONNECT /"  
  this.writer.write(("CONNECT / HTTP/1.0\r\nHost:" + str + "\r\nAge  
unnelVpnService.getPassword()).getBytes(), 2) + "\r\n\r\n").getByt  
pn_client/src/base$
```

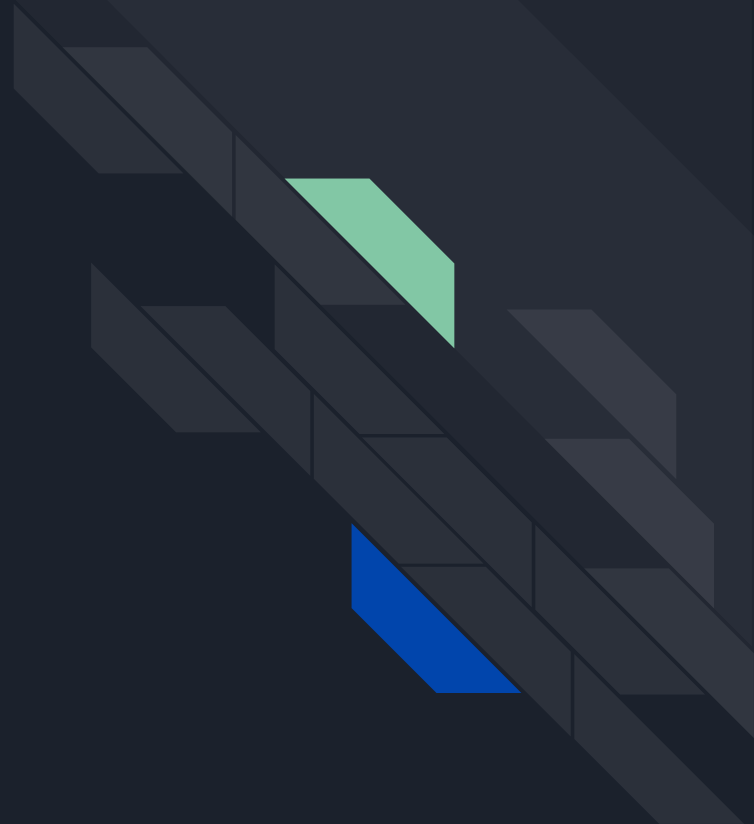


CVE-2025-10547 introduction

- Que se passe t-il lorsque l'utilisateur cherche à se connecter à un script CGI (CONNECT /cgi-bin/Activate.cgi)
- Rien de bon 🐈

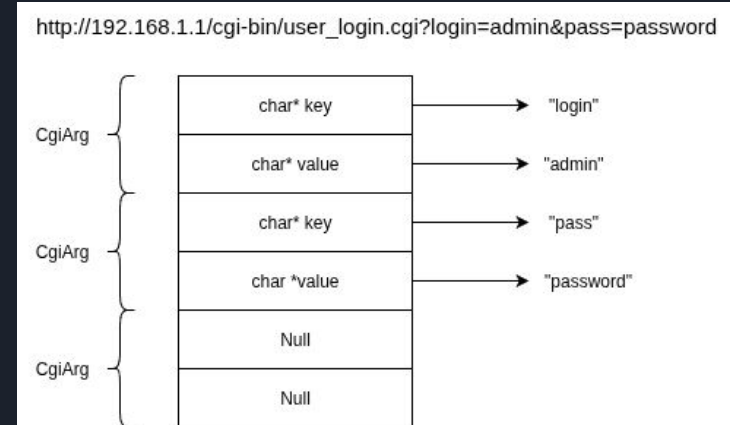
```
lighttpd con service_type: 28 from 192.168.1.10 :50820 --> 192.168.1.10
  dump_backtrace: fp:0x475fa520, pc:0x405f6650
  Call trace: 0x405f6650 0x405d2cf4 0x4133dbb4 0x41431f54 0x401138d0 0x401138d0
SLAB_PANIC:Wrong address range for kfree(0x475fa600) at file:, line:
  dump_backtrace: fp:0x475fa520, pc:0x405f6650
  Call trace: 0x405f6650 0x405d2cf4 0x4133dbb4 0x41431f54 0x401138d0 0x401138d0
SLAB_PANIC:Wrong address range for kfree(0x4011385c) at file:, line:
  dump_backtrace: fp:0x475fa4c0, pc:0x405f6650
  Call trace: 0x405f6650 0x405d2244 0x405d235c 0x405d2f38 0x4133dbb4 0x4133dbb4
SLAB_PANIC:Wrong object pointer <4c461901> at file:NULL line:350! kfree
  dump_backtrace: fp:0x475fa4c0, pc:0x405f6650
  Call trace: 0x405f6650 0x405d2244 0x405d23d8 0x405d2f38 0x4133dbb4 0x4133dbb4
```

CVE-2025-10547
root cause analysis



CVE-2025-10547 root cause analysis

- Fonction s'occupant de l'analyse syntaxique des arguments envoyés en GET ou en POST
- Les résultats sont stockés dans un tableau de pointeurs de chaînes de caractères alternant *key et *value



CVE-2025-10547 root cause analysis

- Fonction s'occupant de l'analyse syntaxique des arguments envoyés en GET ou en POST
- Problème lorsque la méthode HTTP est la méthode CONNECT, les arguments ne sont pas analysés et le tableau n'est jamais initialisé

```
iVar1 = get_cgi_env_array_value("REQUEST_METHOD",param_3);
if (iVar1 == 0) {
    return 0;
}
iVar2 = safe_strncmp(iVar1,"GET");
if (iVar2 == 0) {
    // parse GET params
}
else {
    iVar1 = safe_strncmp(iVar1,"POST");
    if (iVar1 != 0) {
        return 0; // if neither POST or GET request
    }
    // parse POST params
}
args[uVar8].key = (char *)0x0;
return 1;
```

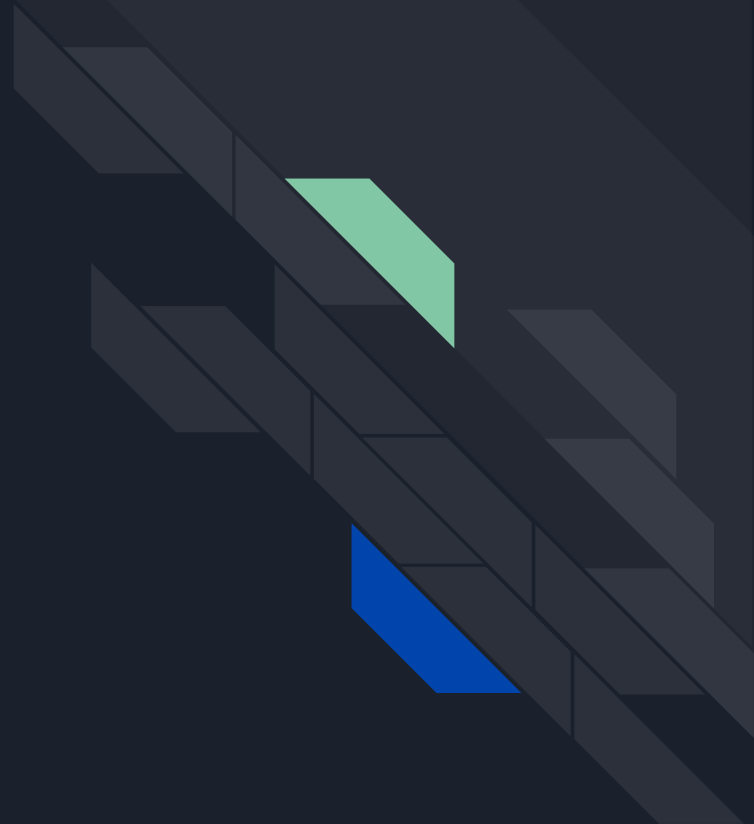
CVE-2025-10547 root cause analysis

- La fonction s'occupant de libérer le tableau d'arguments précédemment construit itère et libère tant que la valeur NULL n'est pas présente dans une clef (fin du tableau)
- Si la valeur NULL n'a jamais été placée dans le tableau, l'application va libérer des valeurs n'ayant pas été initialisées !
- CWE-457: Use of Uninitialized Variable

```
void free_inputs_4133db7c(CgiArg *cgi_params)
{
    int i;

    for (i = 0; cgi_params[i].key != (char *)0x0; i =
        kfree(cgi_params[i].key, 0x15e);
        cgi_params[i].key = (char *)0x0;
    }
    return;
}
```

CVE-2025-10547
free anywhere





CVE-2025-10547 free anywhere

- Possibilité de free des pointeurs non initialisés dans la stack
- A première vue les données ne sont pas maîtrisables par l'utilisateur
- Plusieurs observations
 - En utilisant BurpSuite, certaines données semblaient être contrôlables
 - En utilisant le endpoint `/cgi-bin/user_login.cgi` l'utilisation de la vulnérabilité ne fait pas crash le firmware !
 - Les données que l'utilisateur pouvait manipuler n'arrivaient pas de façon certaine ou prévisibles

CVE-2025-10547 free anywhere

- Tester les différents algorithmes de chiffrement TLS afin de voir si les données manipulables arrivaient plus souvent
- TLS 1.3 (qu'il faut forcer dans ce cas précis) obtient de bons résultats sans toutefois être prévisible
- Les valeurs maîtrisables sont mises dans la pile par l'ordonnanceur (scheduler) depuis les données envoyées dans la requête

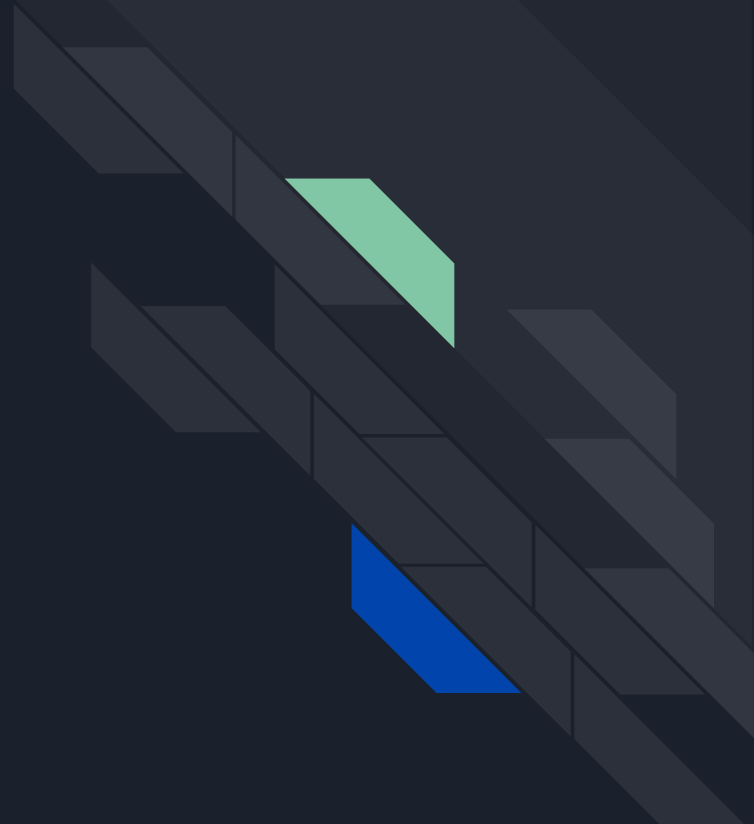
```
dump_backtrace: fp:0x475f9ee0, pc:0x405f6650
Call trace: 0x405f6650 0x405d2cf4 0x4133dbb4 0x41474b50 0x401138
SLAB_PANIC:Wrong address range for kfree(0x14e0a3b1) at file:, 1

dump_backtrace: fp:0x475f9ee0, pc:0x405f6650
Call trace: 0x405f6650 0x405d2cf4 0x4133dbb4 0x41474b50 0x401138
SLAB_PANIC:Wrong address range for kfree(0x22222222) at file:, 1

dump_backtrace: fp:0x475f9ee0, pc:0x405f6650
Call trace: 0x405f6650 0x405d2cf4 0x4133dbb4 0x41474b50 0x401138
SLAB_PANIC:Wrong address range for kfree(0x22222222) at file:, 1

dump_backtrace: fp:0x475f9ee0, pc:0x405f6650
Call trace: 0x405f6650 0x405d2cf4 0x4133dbb4 0x41474b50 0x401138
SLAB_PANIC:Wrong address range for kfree(0x2b1f0e67) at file:, 1
```

CVE-2025-10547
Heap spray



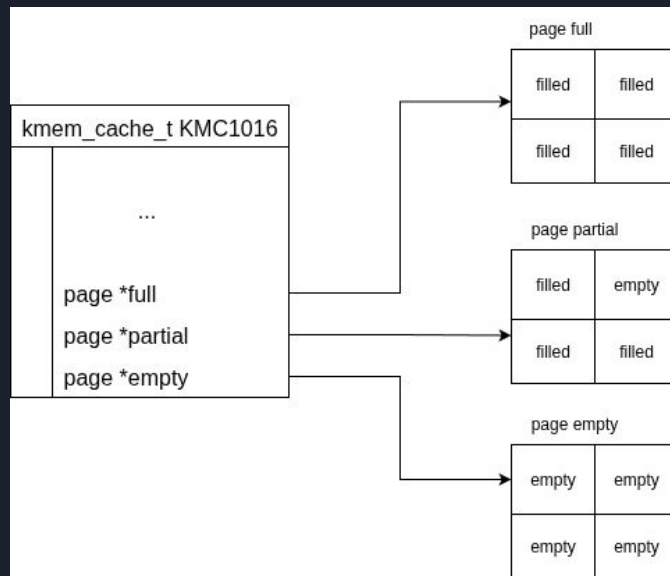


CVE-2025-10547 Heap spray

- L'allocateur mémoire est propre au firmware
- Trouver un moyen d'envoyer des chunks de données contrôlées avec une taille choisie par l'attaquant
- S'assurer que les buffers alloués par kmalloc soient contigus en mémoire afin de former une région de taille suffisante, permettant de cibler correctement un buffer avec la primitive de type free anywhere

CVE-2025-10547 Heap spray

- Les big chunks ($> 0x1000$) sont alloués les uns à la suite des autres
- Le premier big chunk créé est appelé `dynamic_mem_pool` va servir à allouer les petits chunks ($< 0x1000$)
- Les petits chunks sont référencés dans les `kmem_cache` qui créent des pools de buffer situés dans des pages
- Comportement similaire au slab allocator du kernel





CVE-2025-10547 Heap spray

- Utiliser le Transfer-Encoding chunked de HTTP pour créer des données maîtrisées avec une taille maîtrisée
- Impossible car lighttpd est en version 1.4.48. Cette version est buggée et ne permet pas d'envoyer des chunk de plus de 8 octets
- Corrigé en version 1.4.49

👤 [Glenn Strauss](#), 🕒 7 years ago (January 12, 2018 at 7:43 AM)

[core] fix POST with chunked request body (fixes #2854)

(thx the_jk)

x-ref:

"chunked transfer encoding in request body only works for tiny chunks"

<https://redmine.lighttpd.net/issues/2854>

2 files changed, 18 insertions(+), 2 deletions(-)

🔗 [dc1675e](#) | 🏠 [Open on GitHub](#)

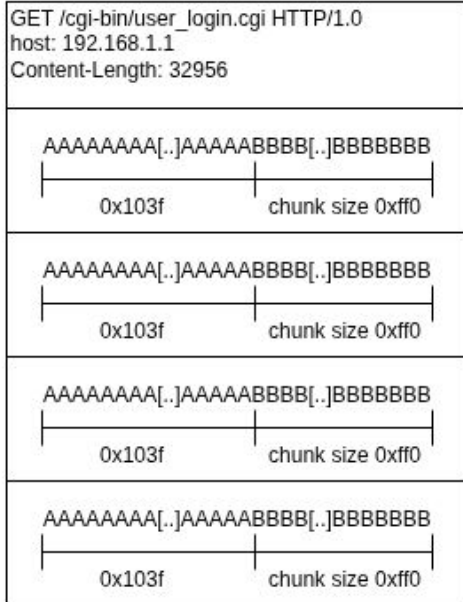
CVE-2025-10547 Heap spray

- SSL_pending retourne le nombre d'octets présents dans un SSL record déchiffré n'ayant pas été lus par un appel à SSL_read
- Un SSL record est un paquet de données chiffrées
- Nécessité d'avoir un premier appel à SSL_read afin de déchiffrer la totalité du SSL record

```
compile: connection_read_cq_ssl_4100c884 - (soho2962.bin)
if (cq == con->read_queue) {
    FUN_4100ac38(con->field50_0x1f0);
    do {
        if (con->page_index == 0x1b5c) {
            FUN_40ffb6c0(con);
            FUN_40ffb8c8(con->read_queue,mem,&local_20);
        }
        else {
            cq_00 = con->read_queue;
            alloc_size = SSL_pending((SSL *)*local_4);
            chunkqueue_get_memory(cq_00,mem,&local_20,0,alloc_size);
        }
        local_8 = SSL_read((SSL *)*local_4,mem[0],local_20);
        if (local_8 < 1) {
            FUN_40ffa500(con->read_queue,0);
        }
    }
}
```

CVE-2025-10547 Heap spray

Request sent

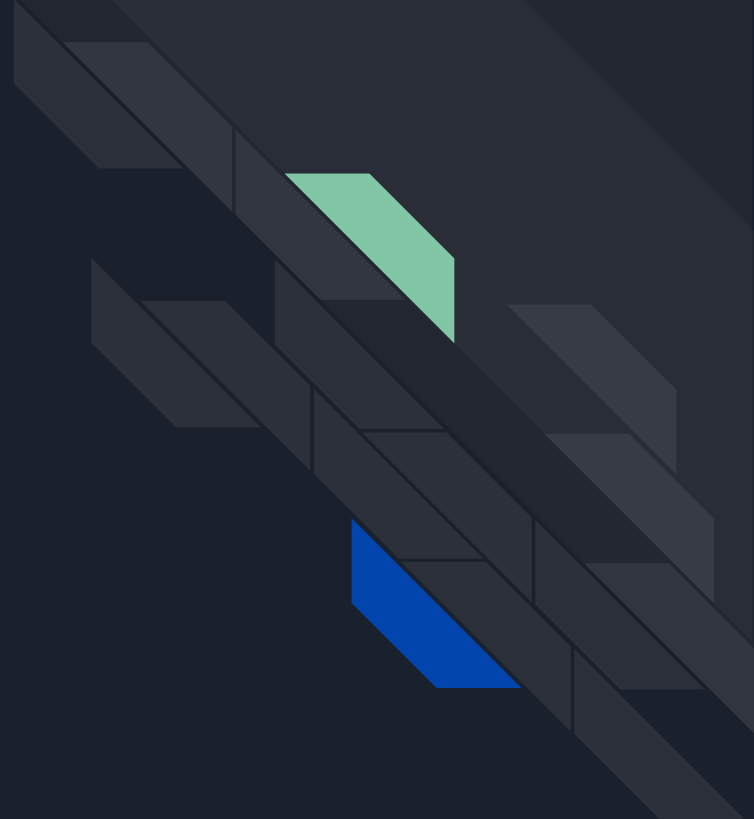


Result in the heap



$$32956 = (0x103f + 0xff0) * 4$$

CVE-2025-10547
Exécution de code



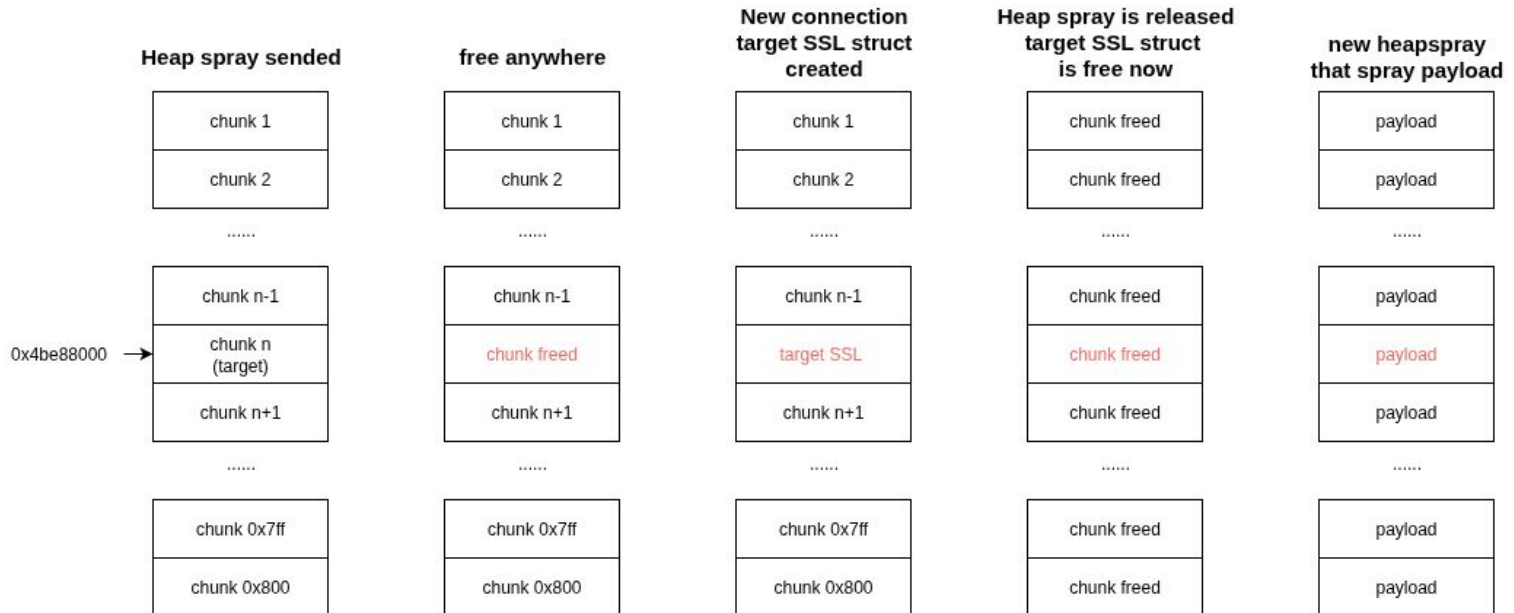


CVE-2025-10547 Exécution de code

- La primitive de type free anywhere permet de libérer un chunk
- Le heap spray permet de créer une cible suffisamment grande pour utiliser le free anywhere
- Une fois le chunk libéré, on se retrouve dans la situation d'un "use after free"
- La structure SSL pourra remplacer le chunk qui a été free
 - Elle est dans le kmem_cache_t 0x1000
 - Elle possède des pointeurs de fonction qui pourront être utilisés pour exécuter du code
- Le firmware ne possède aucune protection mémoire (ASLR, NX, PIE ...)

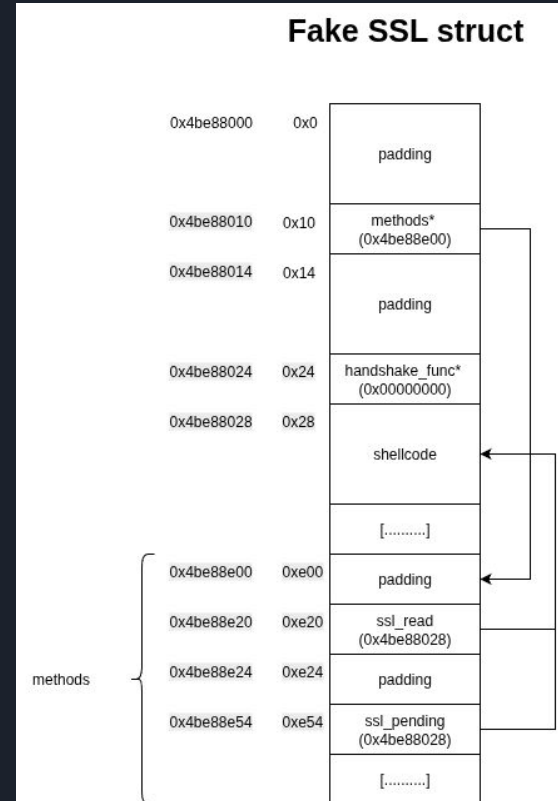
CVE-2025-10547 Exécution de code

Runtime Memory Exploit Flow

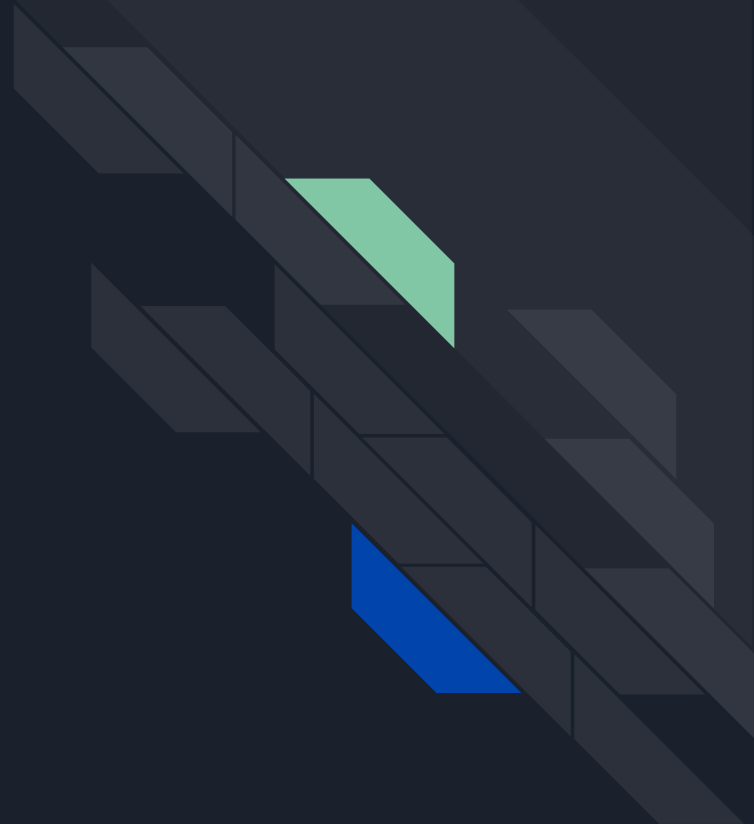


CVE-2025-10547 Exécution de code

- L'attribut `methods*` pointe vers un tableau de handlers
- Les handlers `ssl_read` et `ssl_pending` sont détournés afin de les faire pointer sur le shellcode

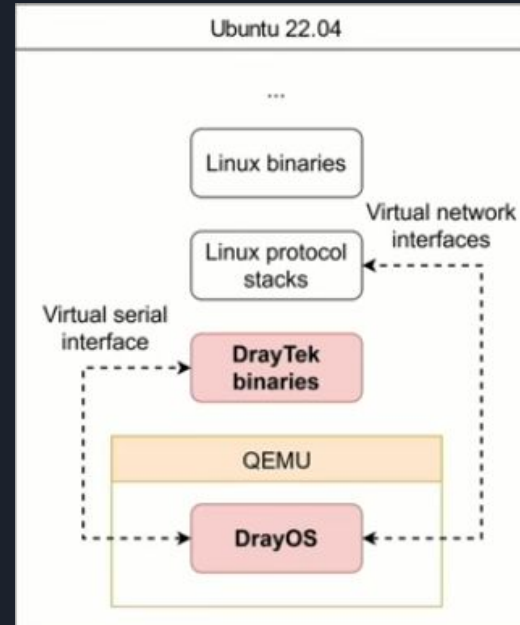


CVE-2025-10547
Rebond sur l'hôte



CVE-2025-10547 Rebond sur l'hôte

- L'exécution de code est située dans DrayOS
- Plusieurs scripts de l'hôte peuvent être appelés depuis DrayOS
- Un script a été trouvé permettant une écriture de fichier arbitraire



CVE-2025-10547 Rebond sur l'hôte

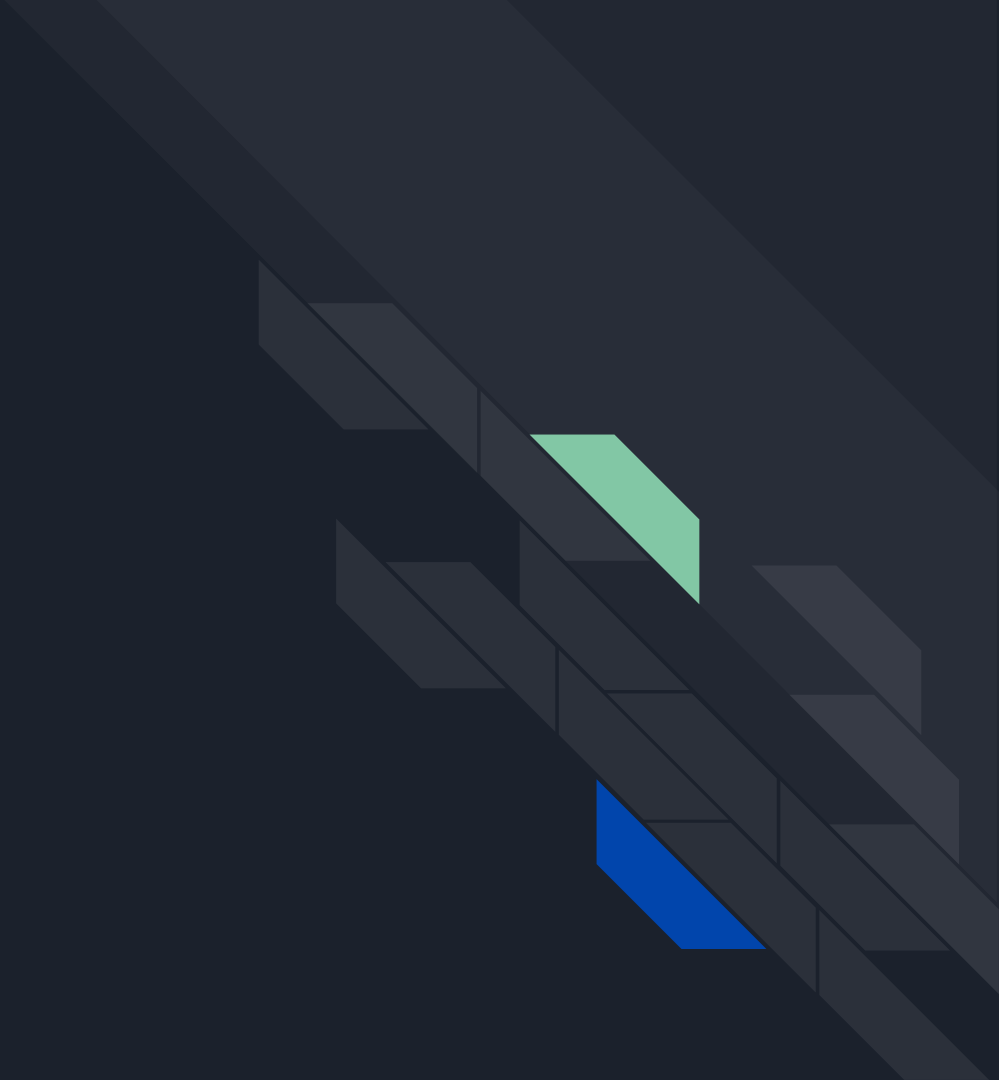
```
$ uffssave x
runcommand > $ uffssave
 1  #!/bin/sh
 2
 3  source /draytek/drayapp/scripts/command_func.sh
 4
 5  frmcmd=$1
 6  frmcnt=$2
 7  frmdst=$3
 8
 9  echo "uffssave $frmcmd $frmcnt $frmdst" >> /var/log/drayos/current
10
11  if [ "$frmcmd" = "uffs" ]; then
12      if [ ! -d /tmp/uffs ]; then
13          mkdir -p /tmp/uffs
14      fi
15      frmfcnt=$(ls -l /tmp/uffs | wc -l)
16      if [ $frmcnt = $frmfcnt ]; then
17          echo "cat $(ls -l -x /tmp/uffs/ | sed -e 's/u\/\tmp\/uffs\/u/g') > $frmdst" >> /var/log/drayos/current
18          cat $(ls -l -x /tmp/uffs/ | sed -e 's/u\/\tmp\/uffs\/u/g') > $frmdst
19          rm /tmp/uffs/*
20      fi
21  fi
```



CVE-2025-10547 Rebond sur l'hôte

- Le fichier créé ne possède pas la permission en exécution
- Modification du script `/draytek/drayapp/scripts/command_func.sh` qui est sourcé dans les scripts
- Appel du script `handle_littlebell` qui en sourçant `command_func.sh` va exécuter notre exploit

DEMO



Des questions ?

