Recherche de vulnérabilités à l’aide d’outil d’analyse automatique de code

SYNACKTIV

A propos

0xMitsurugi

Security Expert

Synacktiv a pour objectif d’aider les entreprises à évaluer et améliorer le niveau de sécurité de leur système d'information.
La société a été fondée en 2012 par deux experts en sécurité informatique. Ils n’ont de cesse depuis ce jour de faire de Synacktiv la référence française en matière de sécurité offensive.

SYNACKTIV RECRUTE!

$ curl github.com/project | ./vazy.sh -o 0days

C’était l’idée à la base

weggli

Brisez les codes avec weggli

L'outil

Rapide
Simple à employer

Syntaxe

Connaît le C
Connaît le C++
Simple à lire/modifier

Oui

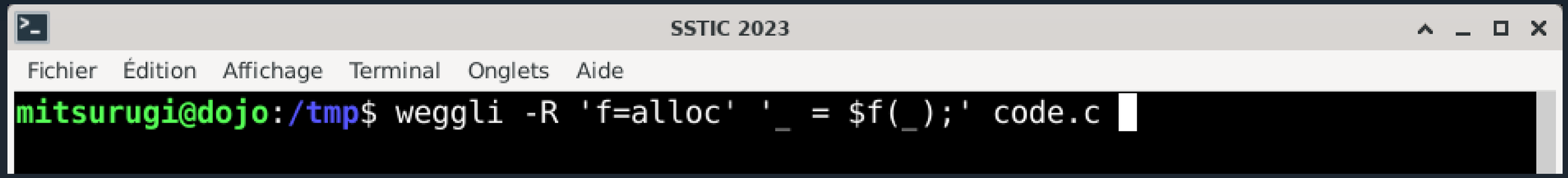
Un supergrep
Outil mis à jour
Peu de patterns publiques

Mais

"juste" un supergrep
Sans patterns, l'outil n'est rien

Un peu de syntaxe

Unleash the power of Incredible Weggli! Grab your copy now!

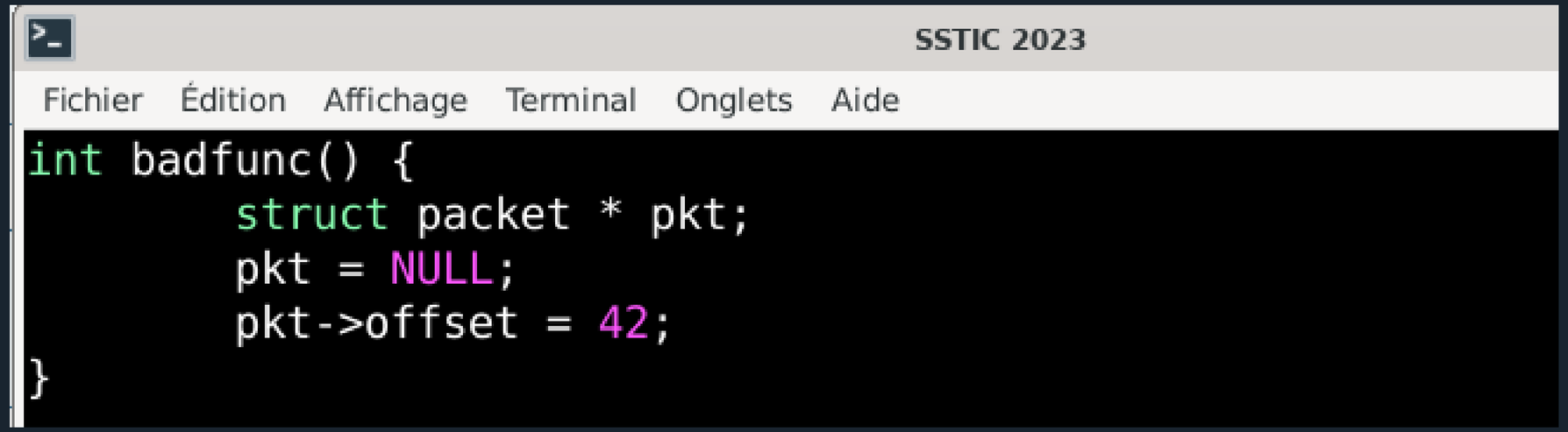


Simplicité d'écriture

Permet rapidement d'écrire des requêtes en mode essai/erreur
Travaille récursivement sur des dossiers
Fonctionne sur du décompilé IDA
Fonctionne sur du code incomplet/qui ne compile pas

A la chasse aux vulns

Il y a le bon chasseur de vuln et le mauvais chasseur...



- **Ecriture des patterns**
  - Ne pas chercher des vulns, mais des *motifs* de vulns
  - Toujours avoir du code à tester
  - Ne pas hésiter à lancer sur des grosses codebases

La bonne pattern

... mais là, c'est un bon chasseur de vuln

SSTIC 2023

Fichier Édition Affichage Terminal Onglets Aide

```
char * src;
char * dst;
        (....)
strcpy(dst, src);      //bad coding!
```

SSTIC 2023

inal Onglets Aide

```
char * src;
char * dst;
        (....)
strncpy(dst, src, strlen(src));      //better ?
```

Sad but true

Pattern matching limité à l'échelle d'une fonction
Pas de comparateur numérique
Impossible de couvrir toutes les vulns
Equilibre difficile à atteindre quantité/faux positifs

Les patterns

Donnez moi 6h pour abattre un arbre, j'en passe 4 à aiguiser ma hache

Patterns dispos sur le github synacktiv

Autodocumentées
Tirées d'expérience, de CTF, de CWE et des vulns rencontrées dans la vraie vie

Fonctions dangereuses

gets, system, alloca, getenv, strcpy/strncpy, format string, etc..

Stack

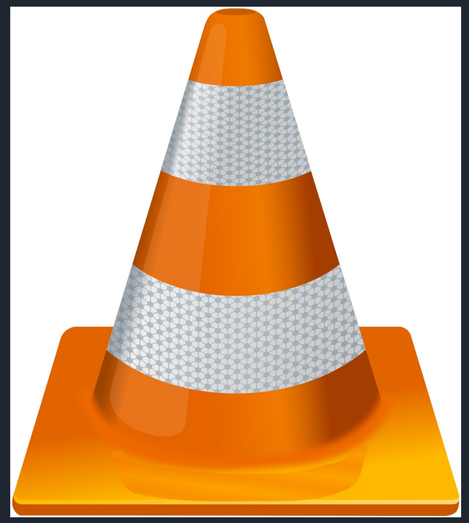
Copies faites sur des buffer de stack, utilisation d'arrays sur la stack

Malloc

Overflow lors d'un calcul de taille à malloc

Des cibles

Ready! Aim! Fire!



Qemu

Parcequ'il fallait bien montrer des faux positifs

```
static void vnc_clipboard_provide(VncState *vs,
                                  QemuClipboardInfo *info,
..
    default:
        return;
    }
    flags |= VNC_CLIPBOARD_PROVIDE;

    buf = g_malloc(info->types[type].size + 4);
    buf[0] = (info->types[type].size >> 24) & 0xff;
    buf[1] = (info->types[type].size >> 16) & 0xff;
    buf[2] = (info->types[type].size >>  8) & 0xff;
    buf[3] = (info->types[type].size >>  0) & 0xff;
    memcpy(buf + 4, info->types[type].data, info->types[type].size);
    zbuf = deflate_buffer(buf, info->types[type].size + 4, &zsize);
    if (!zbuf) {
        return;
    }

..
```

```
$buf = $alloc($a+$b);
$cpy(_,_,$a);
```

Samba

"Samba Safety Shuffle: A Minor Bump, Secure Your Steps!"

```
$buf = $alloc($a*_);
```

```
samba-4.16.4/nsswitch/winbind_nss_aix.c +349
static char *wb_aix_getgrset(char *user)
{
      (...)
        ret = winbindd_request_response(NULL, WINBINDD_GETGROUPS,
                                        &request, &response);

      (...)
        num_gids = response.data.num_entries;                    //extrait de la réponse
        gid_list = (gid_t *)response.extra_data.data;

        /* allocate a space large enough to contruct the string */
        tmpbuf = malloc(num_gids*12);                            //overflow possible et petite allocation
        if (!tmpbuf) {
                return NULL;
        }

        for (idx=i=0; i < num_gids-1; i++) {
                idx += sprintf(tmpbuf+idx, "%u,", gid_list[i]);  //boum
      (...)
```

VLC

Stay tuned! VLC vuln incoming!

```
$size = _*_;
_ = $alloc($size);
```

```
static rfbBool mallocFrameBufferHandler( rfbClient* p_client )
{
..
        p_client->format.blueMax = videofmt.i_bmask >> videofmt.i_lbshift;
        video_format_Clean( &videofmt );
    }

    /* Set up framebuffer */
    p_sys->i_framebuffersize = i_width * i_height * i_depth / 8;

    /* Reuse unsent block */
    if ( p_sys->p_block )
        p_sys->p_block = block_Realloc( p_sys->p_block, 0, p_sys->i_framebuffersize );
    else
        p_sys->p_block = block_Alloc( p_sys->i_framebuffersize );
```

CVE-2022-41325

Director's Cut: Update VLC, Protect Your Reel Experience!

Primitive d'exploitation très puissante

Ecriture choisie d'octets au delà du buffer
Exploit trivial sans ASLR (réécriture de la stack depuis le heap)
Réponse très rapide de l'équipe VLC, patch dans la journée

Use CVE-2022-41325.

- --
CVE Assignment Team
M/S M300, 202 Burlington Road, Bedford, MA 01730 USA
[A PGP key is available for encrypted communications at
https://cve.mitre.org/cve/request_id.html]

Leçons apprises

Du bon, du moins bon, mais toujours des leçons pour l'avenir!

Des surfaces d'attaques inattendues

VNC pour VLC et qemu
Sous titres pour VLC
Format d'images et integer overflow, etc...

L'outil est limité

Beaucoup de faux positifs (mais c'est lié au choix d'écriture des patterns)
Besoin de retravailler les patterns selon les projets (nommage, macro, etc..)
Limité dans la recherche de motifs compliqués

Mais très pratique

Achievement unlock : CVE-2022-41325
Très efficace pour la pêche au gros
Permet d'explorer rapidement un soft

A propos

Merci de votre attention,

SSTIC 2023