T-Brop
Taint-Based Return Oriented Programming

Colas Le Guernic

DGA Maîtrise de l'Information

Univ. Rennes, Inria, CNRS, IRISA

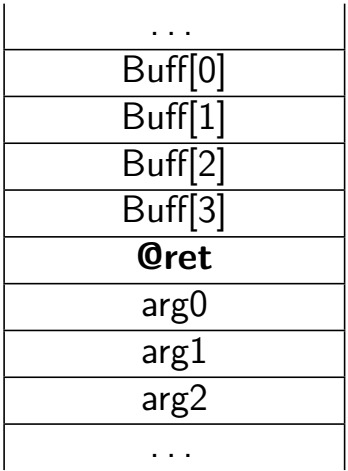
François Khourbiga

~~DGA Maîtrise de l'Information~~

Orange Cyberdéfense

SSTIC 2018

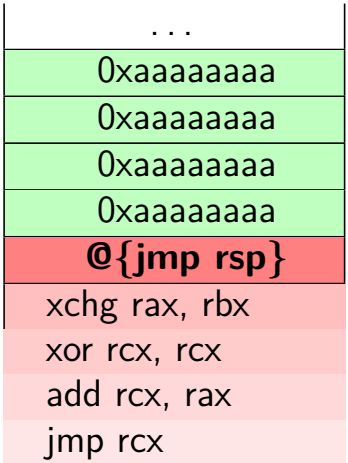
Return Oriented Programming



Débordement de tampon sur la pile

- injection de code

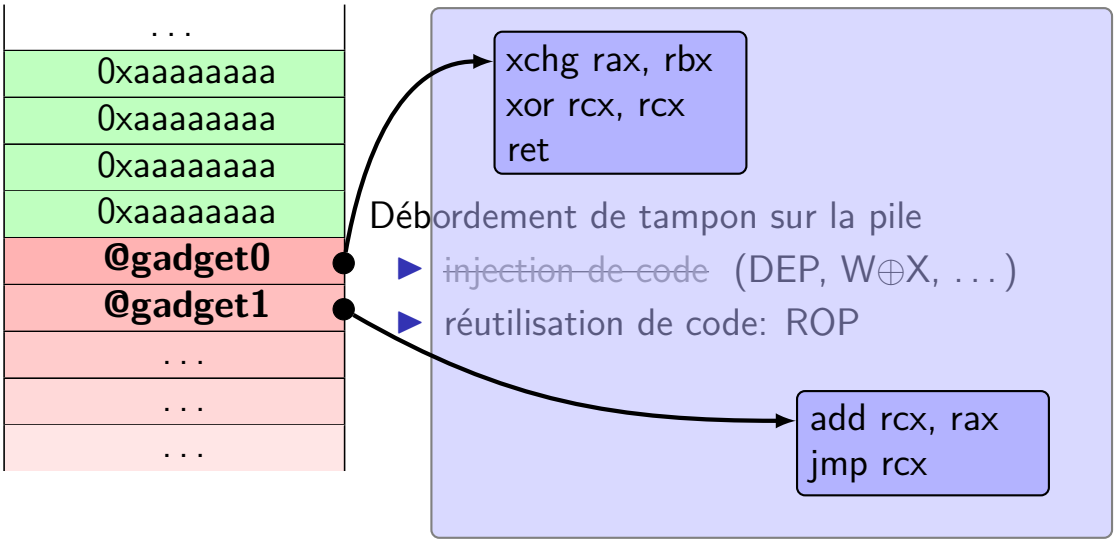
Return Oriented Programming



Débordement de tampon sur la pile

- injection de code

Return Oriented Programming



Pourquoi?

#ropchain

Pourquoi?

~~Auditeur incapable de contourner le DEP~~
~~$\Longrightarrow$ DEP est suffisant~~

Convaincre de l'importance des contre-mesures modernes (sandbox, CFI, . . .)

Rarement suffisant mais mieux que rien

État de l'Art: Deux Approches Principales

Approche Syntaxique

- liste les gadgets
- [+] Rapide
- [−] Recherche par regexp
- Exemple: **RP++**

Approche Symbolique

- Calcul relation I/O symbolique
- [−] Couteux
- [+] Filtrage expressif et précis
- Exemple: **angrop**

Chaînage (semi-)automatique à base de templates (et Z3)

État de l'Art: Deux Approches Principales

Approche Syntaxique

- liste les gadget
- [+] Rapide
- [−] Recherche
- Exemple: **RP**

Approche Symbolique

- Calcul relation I/O symbolique
- [+] Précis
- [+] Filtrage expressif et précis
- Exemple: **Angrop**

T-Brop

- Calcul dépendances I/O
- Compromis Symb./Syn.
- [∼] Plus rapide que Symb.
- [∼] Plus expressif que Syn.

Chaînage (semi-)automatique à base de templates (et Z3)

Démo: Hypothèses

L'analyste contrôle:

- un buffer pointé par [rsp+8] dont il connaît l'adresse
- la cible du prochain saut

L'analyste recherche un stack pivot:

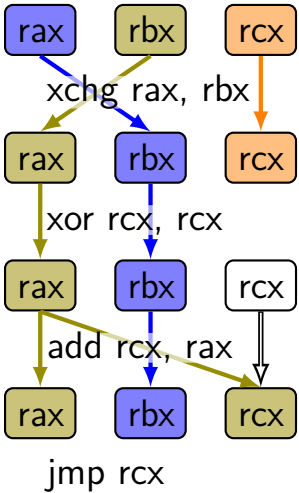
- rsp $\leftarrow$ [rsp+8]
- ou: rsp $\leftarrow$ [[rsp+8] + ??]

Démo: Résultats

- RP++
- angrop
- T-Brop

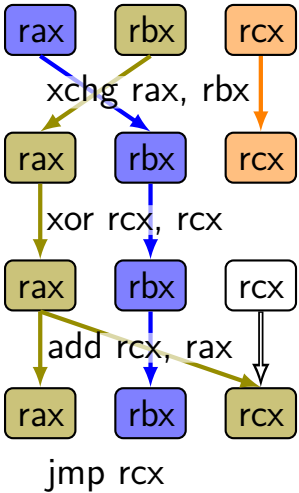
```
mov rcx, qword ptr [rsp + 8];
mov byte ptr [rsp], dl;
mov rax, qword ptr [rcx];
mov rdi, rcx;
call qword ptr [rax + 0x48];

mov rsp, rcx;
ret;
```

Approche basée sur la teinte



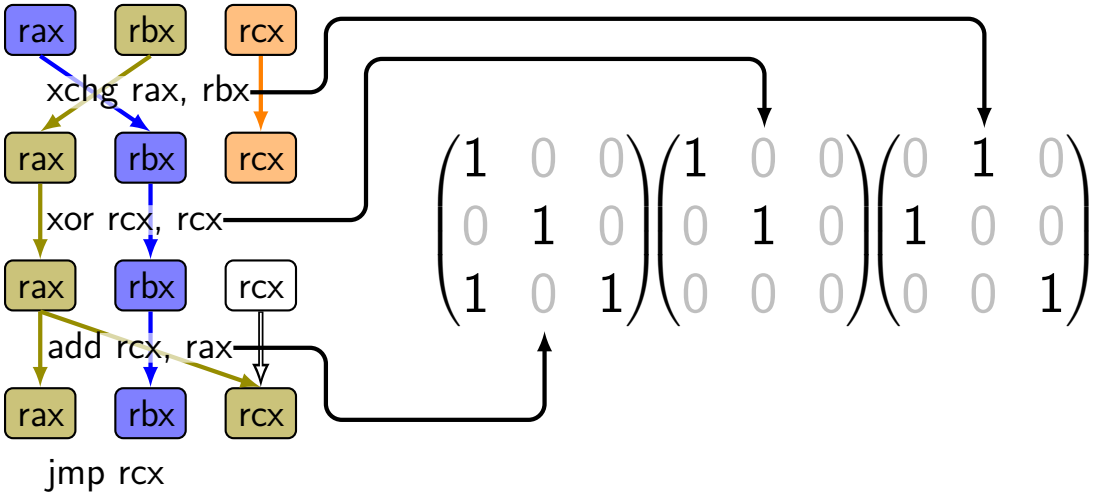
out \ in	rax	rbx	rcx
rax	·	↵	·
rbx	↵	·	·
rcx	·	↵	·

Approche basée sur la teinte

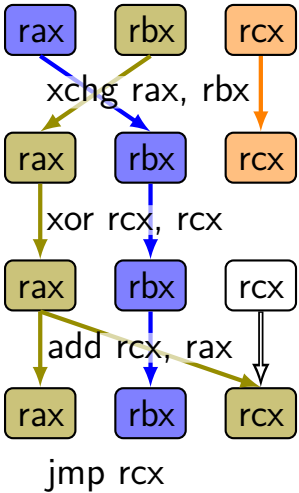


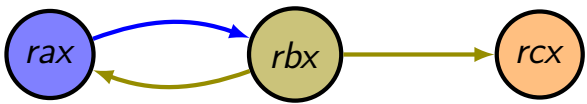
$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

Approche basée sur la teinte



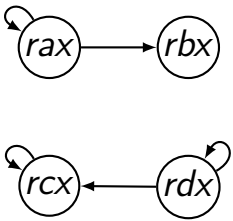
Approche basée sur la teinte

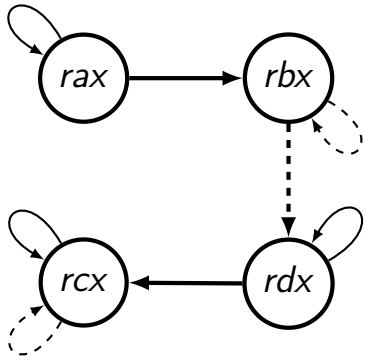




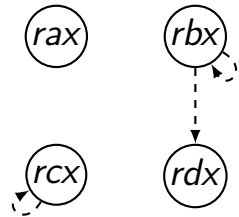
Superposition de Gadgets

mov rbx, rax
xor rcx, rdx
ret





push rbx
xor rax, rax
pop rdx
ret



Superposition de Gadgets

```
mov rbx, rax
xor rcx, rdx
ret
```

$$M_1 + M_2$$

```
push rbx
xor rax, rax
pop rdx
ret
```

M_1

M_2

Superposition et Enchaînement

- $M_1 + M_2$: gadget 1 ou 2
- I: aucun gadget (nop)
- Un gadget arbitraire de G:

$$\mathcal{M}_G = \sum_{g \in G} M_g$$

$$(I + \mathcal{M}_G)^\infty$$

- $M_2 M_1$: gadget 1 puis 2

Effet d'une chaîne dans G de longueur arbitraire

Dépendance absente $\Longrightarrow$ inutile de chercher une chaîne

Condition d'Enchaînement

```
mov [rsp + 0x48], rsi
mov [rsp + 0x50], rax
add rsp, 0x48
pop rdi
ret
```

```
++++ DepMatrix ++++
rflags <--- rsp
rdi <--- rsi
deref <--- rsp
stack_{-2} <--- rsi
stack_{-1} <--- rax

++++ chainCond ++++
rax
```

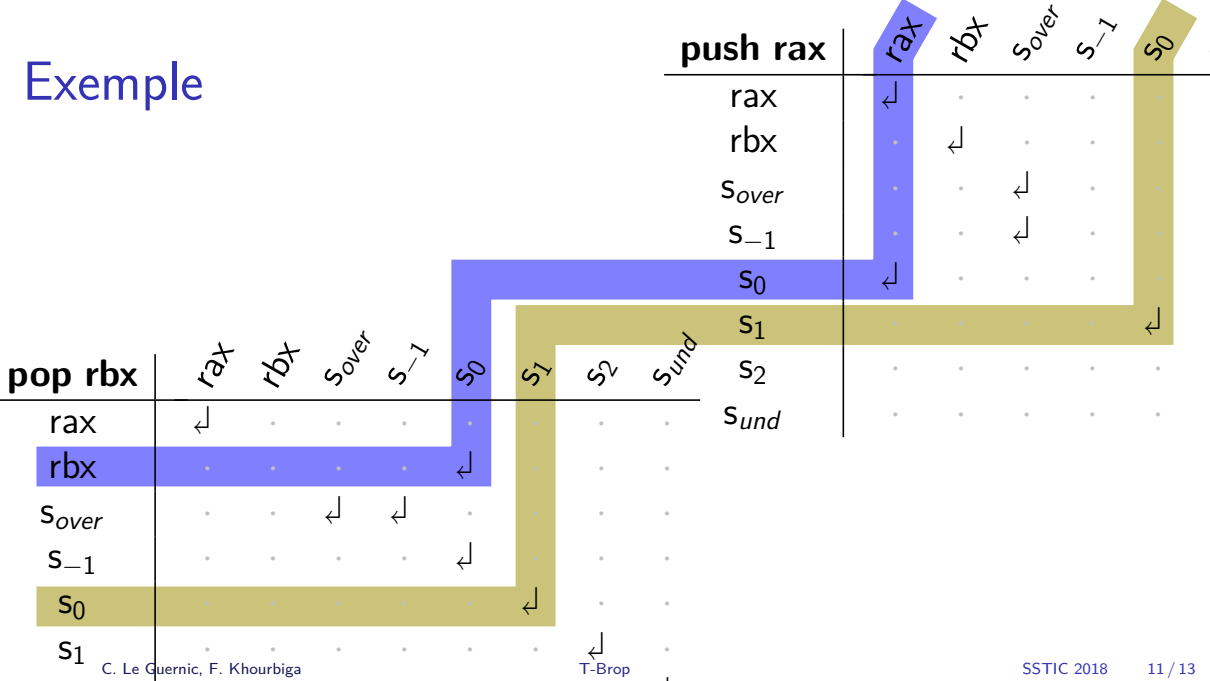
Chaîne (M_1, C_1) puis (M_2, C_2): $(M_2M_1, C_2M_1 + C_1)$

Gestion de la Mémoire et de la Pile

Deux approches:

- **Correcte** (pas de FN): un pseudo-registre mem
- **Utile** (moins de FP):
  deux pseudo-registres mem_r et mem_w
  un pseudo-registre pour chaque cellule de la (pseudo-)pile
  deux pour le haut de pile, deux pour le bas de pile

Exemple


push rax
rax
rbx
s_{over}
s_{-1}
s_0
s_1
s_2
s_{und}
pop rbx
rax
rbx
s_{over}
s_{-1}
s_0
s_1
s_2
s_{und}

PoC: Implémentation

python3 + capstone (next) + SciPy + NumPy

Binaire analysé		Temps de traitement en minutes:secondes				
Nom	Taille (ko)	rp++	ROPgadget	T-Brop	angrop	nrop
fstab-decode	6	0:00.00	0:00.15	0:00.90	*0:07.66*	0:24.92
java	6	0:00.00	0:00.15	0:00.77	0:07.56	0:33.18
fgconsole	10	0:00.01	0:00.18	0:01.13	0:12.75	1:42.91
nisdomainname	14	0:00.01	0:00.16	0:01.30	0:16.25	3:07.10
openvt	19	0:00.02	0:00.20	0:01.54	0:18.04	5:36.12
echo	31	0:00.06	0:00.30	0:03.67	0:59.79	13:00.73
rmdir	39	0:00.10	0:00.39	0:05.00	*1:11.10*	11:22.58
touch	63	0:00.13	0:00.51	0:07.02	1:38.85	20:11.61
kmod	151	0:00.35	0:01.23	0:18.73	3:59.10	42:00.60
tar	375	0:01.10	0:03.51	0:58.47	*2:45.70*	
c++	898	0:01.81	0:04.86	1:44.97	*15:40.45*	
rbash	1013	0:03.01	0:11.56	2:38.21	*25:53.19*	
static-sh	1918	0:05.74	0:18.22	9:07.77	*47:01.06*	
python3	4360	0:11.27	0:23.45	17:14.36	*21:12.80*	

Contributions

- Nouvelle approche pour la sélection de gadgets
- Compromis Symbolique/Syntaxique
- Matrice de dépendances, condition d'enchaînement
- Superposition de gadgets
- *Throwaway script*: github.com/DGA-MI-SSI/T-Brop

Perspectives

- Enchaînement semi-automatique $\approx$ plus court chemin
- Interaction avec approches symboliques
- PoC $\longrightarrow$ Tool

Contributions

- Nouvelle approche pour la sélection de gadgets
- Compromis Symbolique/Syntaxique
- Matrice de dépendances, condition d'enchaînement
- Superposition de gadgets
- *Throwaway scrip* T-Brop

Questions?

Perspectives

- Enchaînement semi-au rt chemin
- Interaction avec approches symb
- PoC ⟶ Tool