Robin Marsollier
CONIX CyberSecurity

–

10 April 2018

Who am I?

Robin Marsollier
Twitter: ▸ @rbnctl
robin.marsollier@conix.fr
CERT-Conix

Machoke: CFG-based malware classification

—

BTG: OSINT tool

Machoke: CFG-based malware classification

CERT-Conix tools: Machoke and BTG

Machoke: definitions

1. fuzzy hash

Machoke: definitions

1. fuzzy hash
2. CFG : control flow graph

1. fuzzy hash
2. CFG : control flow graph
3. clusterisation

1. Apparently used by AV
2. Used by academics
3. Used by other actors

1. Apparently used by AV
2. Used by academics
3. Used by other actors
4. few public implementations ...

Machoke: Objectives

1. Get something better than md5/sha* (resistant to small changes inside samples notably, etc.)
2. A fuzzy hash better than good old ssdeep
3. Get a small and independent tool easy to use and deploy at large
4. Let other tools do the clustering

1. Designed by ANSSI, published with Polichombr (https://github.com/ANSSI-FR/polichombr)
2. CFG-based fuzzy hash
3. 2 implementations: Ruby/miasm || Python/IDAPython (Machoc lost in lots of ruby/python/whatever code)

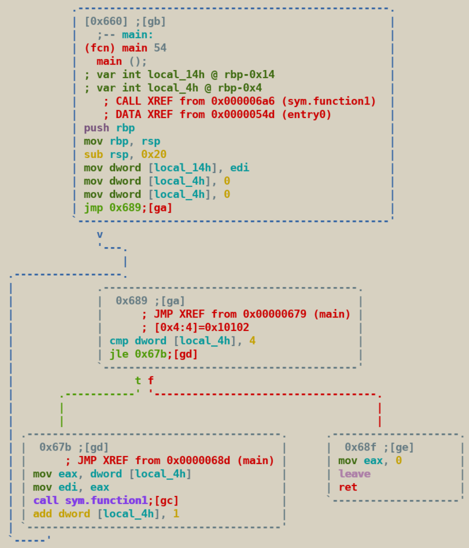
CERT-Conix tools: Machoke and BTG

Machoke: Naming

1. Radare2 + r2pipe
2. Python

CERT-Conix tools: Machoke and BTG

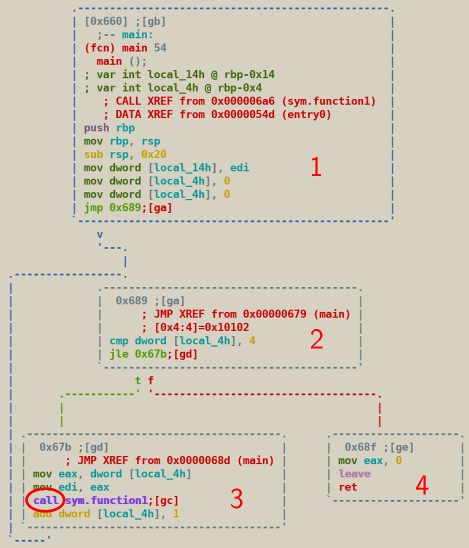
Machoke: algorithm



```
          .-------------------------------------------------.
          | [0x660] ;[gb]                                   |
          |   ;-- main:                                     |
          | (fcn) main 54                                   |
          |   main ();                                      |
          | ; var int local_14h @ rbp-0x14                  |
          | ; var int local_4h @ rbp-0x4                    |
          |    ; CALL XREF from 0x000006a6 (sym.function1)  |
          |    ; DATA XREF from 0x0000054d (entry0)         |
          | push rbp                                        |
          | mov rbp, rsp                                    |
          | sub rsp, 0x20                                   |
          | mov dword [local_14h], edi                      |
          | mov dword [local_4h], 0                         |
          | mov dword [local_4h], 0                         |
          | jmp 0x689;[ga]                                  |
          `-------------------------------------------------'
              v
              '---.
                  |
.-----------------.
|
|               .-----------------------------------------.
|               |  0x689 ;[ga]                            |
|               |       ; JMP XREF from 0x00000679 (main) |
|               |       ; [0x4:4]=0x10102                 |
|               | cmp dword [local_4h], 4                 |
|               | jle 0x67b;[gd]                          |
|               `-----------------------------------------'
|                     t f
|       .------------' '-----------------------------------.
|       |                                                  |
|       |                                                  |
|  .-----------------------------------------.   .---------------------.
|  |  0x67b ;[gd]                            |   |  0x68f ;[ge]        |
|  |       ; JMP XREF from 0x0000068d (main) |   | mov eax, 0          |
|  | mov eax, dword [local_4h]               |   | leave               |
|  | mov edi, eax                            |   | ret                 |
|  | call sym.function1;[gc]                 |   `---------------------'
|  | add dword [local_4h], 1                 |
|  `-----------------------------------------'
`-----'
```

CERT-Conix tools: Machoke and BTG

Machoke: algorithm



```
[0x660] ;[gb]
  ;-- main:
(fcn) main 54
  main ();
; var int local_14h @ rbp-0x14
; var int local_4h @ rbp-0x4
   ; CALL XREF from 0x000006a6 (sym.function1)
   ; DATA XREF from 0x0000054d (entry0)
push rbp
mov rbp, rsp
sub rsp, 0x20
mov dword [local_14h], edi
mov dword [local_4h], 0
mov dword [local_4h], 0
jmp 0x689;[ga]
```

```
 0x689 ;[ga]
      ; JMP XREF from 0x00000679 (main)
      ; [0x4:4]=0x10102
cmp dword [local_4h], 4
jle 0x67b;[gd]
```

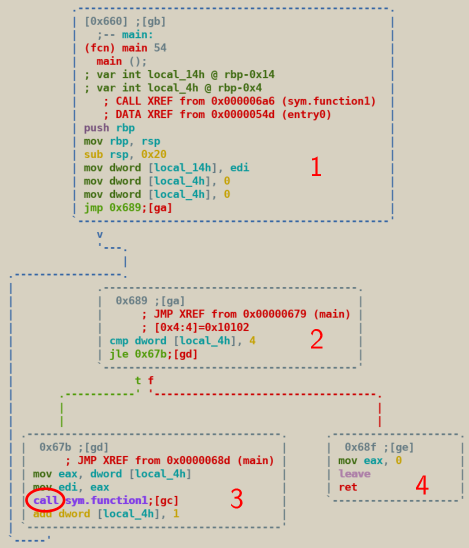
```
 0x67b ;[gd]
      ; JMP XREF from 0x0000068d (main)
mov eax, dword [local_4h]
mov edi, eax
call sym.function1;[gc]
add dword [local_4h], 1
```

```
 0x68f ;[ge]
mov eax, 0
leave
ret
```

1. Blocks and call labelling

CERT-Conix tools: Machoke and BTG

Machoke: algorithm



```
[0x660] ;[gb]
  ;-- main:
(fcn) main 54
  main ();
; var int local_14h @ rbp-0x14
; var int local_4h @ rbp-0x4
   ; CALL XREF from 0x000006a6 (sym.function1)
   ; DATA XREF from 0x0000054d (entry0)
push rbp
mov rbp, rsp
sub rsp, 0x20
mov dword [local_14h], edi
mov dword [local_4h], 0
mov dword [local_4h], 0
jmp 0x689;[ga]
```

```
 0x689 ;[ga]
      ; JMP XREF from 0x00000679 (main)
      ; [0x4:4]=0x10102
cmp dword [local_4h], 4
jle 0x67b;[gd]
```

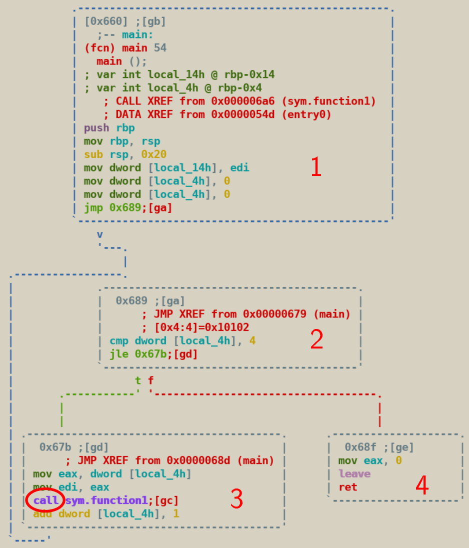
```
 0x67b ;[gd]
      ; JMP XREF from 0x0000068d (main)
mov eax, dword [local_4h]
mov edi, eax
call sym.function1;[gc]
add dword [local_4h], 1
```

```
 0x68f ;[ge]
mov eax, 0
leave
ret
```

1. Blocks and call labelling
2. Translate to text:
   1:2;

CERT-Conix tools: Machoke and BTG

Machoke: algorithm

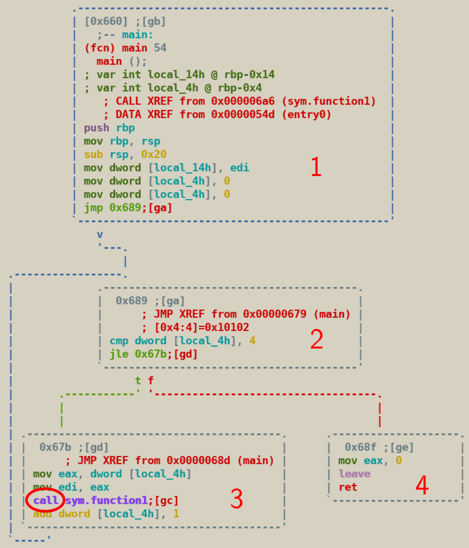


```
                .-------------------------------------------------.
                | [0x660] ;[gb]                                   |
                |   ;-- main:                                     |
                | (fcn) main 54                                   |
                |   main ();                                      |
                | ; var int local_14h @ rbp-0x14                  |
                | ; var int local_4h @ rbp-0x4                    |
                |    ; CALL XREF from 0x000006a6 (sym.function1)  |
                |    ; DATA XREF from 0x0000054d (entry0)         |
                | push rbp                                        |
                | mov rbp, rsp                                    |
                | sub rsp, 0x20                                   |
                | mov dword [local_14h], edi                      |
                | mov dword [local_4h], 0                         |
                | mov dword [local_4h], 0                         |
                | jmp 0x689;[ga]                                  |
                `-------------------------------------------------'
                    v
                    '---.
                        |
 .------------------.
 |
 |                    .-----------------------------------------.
 |                    |  0x689 ;[ga]                            |
 |                    |      ; JMP XREF from 0x00000679 (main)  |
 |                    |      ; [0x4:4]=0x10102                  |
 |                    | cmp dword [local_4h], 4                 |
 |                    | jle 0x67b;[gd]                          |
 |                    `-----------------------------------------'
 |                            t f
 |          .-----------' '-----------------------------------.
 |          |                                                 |
 |          |                                                 |
 | .-----------------------------------------.     .-------------------.
 | |  0x67b ;[gd]                            |     |  0x68f ;[ge]      |
 | |      ; JMP XREF from 0x0000068d (main)  |     | mov eax, 0        |
 | | mov eax, dword [local_4h]               |     | leave             |
 | | mov edi, eax                            |     | ret               |
 | | call sym.function1;[gc]                 |     `-------------------'
 | | add dword [local_4h], 1                 |
 | `-----------------------------------------'
 `-----'
```

1. Blocks and call labelling
2. Translate to text: 1:2;2:3,4;

CERT-Conix tools: Machoke and BTG

Machoke: algorithm



```
[0x660] ;[gb]
  ;-- main:
(fcn) main 54
  main ();
; var int local_14h @ rbp-0x14
; var int local_4h @ rbp-0x4
   ; CALL XREF from 0x000006a6 (sym.function1)
   ; DATA XREF from 0x0000054d (entry0)
push rbp
mov rbp, rsp
sub rsp, 0x20
mov dword [local_14h], edi
mov dword [local_4h], 0
mov dword [local_4h], 0
jmp 0x689;[ga]
```

```
 0x689 ;[ga]
      ; JMP XREF from 0x00000679 (main)
      ; [0x4:4]=0x10102
cmp dword [local_4h], 4
jle 0x67b;[gd]
```

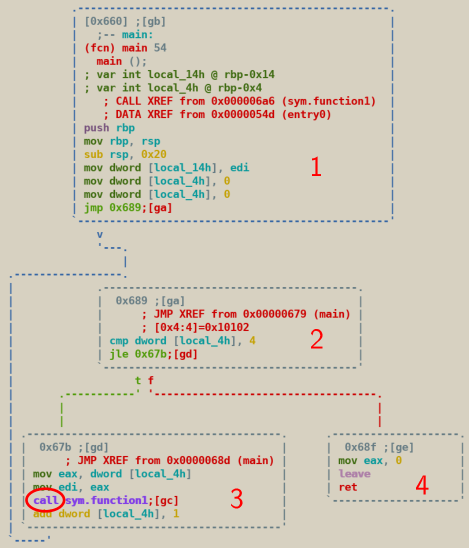
```
 0x67b ;[gd]
      ; JMP XREF from 0x0000068d (main)
mov eax, dword [local_4h]
mov edi, eax
call sym.function1;[gc]
add dword [local_4h], 1
```

```
 0x68f ;[ge]
mov eax, 0
leave
ret
```

1. Blocks and call labelling
2. Translate to text: 1:2;2:3,4;3:c,2;

CERT-Conix tools: Machoke and BTG

Machoke: algorithm



```
[0x660] ;[gb]
  ;-- main:
(fcn) main 54
  main ();
; var int local_14h @ rbp-0x14
; var int local_4h @ rbp-0x4
   ; CALL XREF from 0x000006a6 (sym.function1)
   ; DATA XREF from 0x0000054d (entry0)
push rbp
mov rbp, rsp
sub rsp, 0x20
mov dword [local_14h], edi
mov dword [local_4h], 0
mov dword [local_4h], 0
jmp 0x689;[ga]
```

```
 0x689 ;[ga]
      ; JMP XREF from 0x00000679 (main)
      ; [0x4:4]=0x10102
cmp dword [local_4h], 4
jle 0x67b;[gd]
```

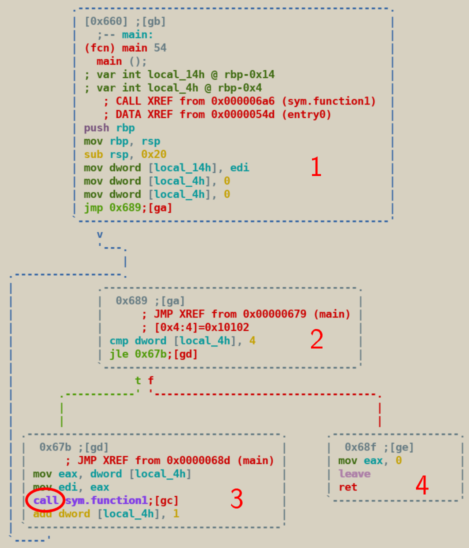
```
 0x67b ;[gd]
      ; JMP XREF from 0x0000068d (main)
mov eax, dword [local_4h]
mov edi, eax
call sym.function1;[gc]
add dword [local_4h], 1
```

```
 0x68f ;[ge]
mov eax, 0
leave
ret
```

1. Blocks and call labelling
2. Translate to text: 1:2;2:3,4;3:c,2;4:;

CERT-Conix tools: Machoke and BTG

Machoke: algorithm



```
[0x660] ;[gb]
  ;-- main:
(fcn) main 54
  main ();
; var int local_14h @ rbp-0x14
; var int local_4h @ rbp-0x4
   ; CALL XREF from 0x000006a6 (sym.function1)
   ; DATA XREF from 0x0000054d (entry0)
push rbp
mov rbp, rsp
sub rsp, 0x20
mov dword [local_14h], edi
mov dword [local_4h], 0
mov dword [local_4h], 0
jmp 0x689;[ga]
```

```
 0x689 ;[ga]
      ; JMP XREF from 0x00000679 (main)
      ; [0x4:4]=0x10102
cmp dword [local_4h], 4
jle 0x67b;[gd]
```

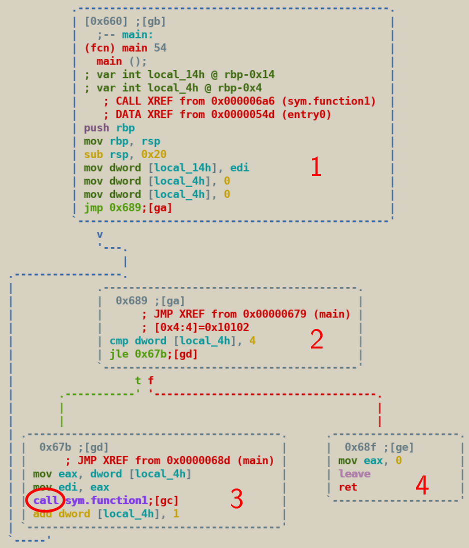
```
 0x67b ;[gd]
      ; JMP XREF from 0x0000068d (main)
mov eax, dword [local_4h]
mov edi, eax
call sym.function1;[gc]
add dword [local_4h], 1
```

```
 0x68f ;[ge]
mov eax, 0
leave
ret
```

1. Blocks and call labelling
2. Translate to text: 1:2;2:3,4;3:c,2;4:;
3. Murmurhash3: e38a5cbb

CERT-Conix tools: Machoke and BTG

Machoke: algorithm



```
[0x660] ;[gb]
  ;-- main:
(fcn) main 54
  main ();
; var int local_14h @ rbp-0x14
; var int local_4h @ rbp-0x4
   ; CALL XREF from 0x000006a6 (sym.function1)
   ; DATA XREF from 0x0000054d (entry0)
push rbp
mov rbp, rsp
sub rsp, 0x20
mov dword [local_14h], edi
mov dword [local_4h], 0
mov dword [local_4h], 0
jmp 0x689;[ga]
```

```
 0x689 ;[ga]
      ; JMP XREF from 0x00000679 (main)
      ; [0x4:4]=0x10102
cmp dword [local_4h], 4
jle 0x67b;[gd]
```

```
 0x67b ;[gd]
      ; JMP XREF from 0x0000068d (main)
mov eax, dword [local_4h]
mov edi, eax
call sym.function1;[gc]
add dword [local_4h], 1
```

```
 0x68f ;[ge]
mov eax, 0
leave
ret
```

1. Blocks and call labelling
2. Translate to text: 1:2;2:3,4;3:c,2;4:;
3. Murmurhash3: e38a5cbb
4. Repeat for each function in sample, concatenate hashes

Machoke: r2 commands used

1. aa
2. ilj
3. aflj
4. agj

Analysis on a (small) collection of 21915 samples:

1. 21915 unique MD5/SHA256 (as expected)

Machoke: Results

Analysis on a (small) collection of 21915 samples:

1. 21915 unique MD5/SHA256 (as expected)
2. 10691 unique ssdeep

Analysis on a (small) collection of 21915 samples:

1. 21915 unique MD5/SHA256 (as expected)
2. 10691 unique ssdeep
3. Only 4674 unique machoke hashes

Demonstration

BTG: OSINT tool

The SOC/CERT analyst stumbles upon an suspicious domain/IP (our observables) and then :

BTG: typical qualification workflow

The SOC/CERT analyst stumbles upon an suspicious domain/IP (our observables) and then :

1. Go on VT and search for the observables

BTG: typical qualification workflow

The SOC/CERT analyst stumbles upon an suspicious domain/IP (our observables) and then :

1. Go on VT and search for the observables
2. Go on MalwareShare and search for the observables

BTG: typical qualification workflow

The SOC/CERT analyst stumbles upon an suspicious domain/IP (our observables) and then :

1. Go on VT and search for the observables
2. Go on MalwareShare and search for the observables
3. Go on MISP and search for the observables
4. Go on Cuckoo and search for the observables

BTG: typical qualification workflow

The SOC/CERT analyst stumbles upon an suspicious domain/IP (our observables) and then :

1. Go on VT and search for the observables
2. Go on MalwareShare and search for the observables
3. Go on MISP and search for the observables
4. Go on Cuckoo and search for the observables
5. Go on PassiveTotal and search for the observables
6. Go on OTX and search for the observables
7. Go on some malware trackers and search for the observables

BTG: typical qualification workflow

The SOC/CERT analyst stumbles upon an suspicious domain/IP (our observables) and then :

1. Go on VT and search for the observables
2. Go on MalwareShare and search for the observables
3. Go on MISP and search for the observables
4. Go on Cuckoo and search for the observables
5. Go on PassiveTotal and search for the observables
6. Go on OTX and search for the observables
7. Go on some malware trackers and search for the observables

So many sources of informations.

BTG: purposes

Help the analyst to get to the websites that contains info/intel about this observable quickly.

BTG: principles

1. Ergonomy (less output is better)
2. Give the right informations to the analyst

BTG: invocation

1. python3 BTG.py [your observable]

BTG: invocation

1. python3 BTG.py [your observable]
2. python3 BTG.py [your observables]

BTG: invocation

1. python3 BTG.py [your observable]
2. python3 BTG.py [your observables]
3. python3 BTG.py [name of file containing your observables]

BTG: output formating

[OSINT Source][Status]{Observable} Link to the ressource

BTG: output formating

[OSINT Source][Status]{Observable} Link to the ressource

Example :

```
[malwareteks][FOUND]{ml314.com} http://hosts-file.malwareteks.com/hosts.txt
```

1. python3
2. 25 modules for 25 OSINT services
3. actively used and maintained @ Conix
4. observables handled: URL, MD5, SHA1, SHA256, SHA512, IPv4, IPv6, domain

Demonstration

Github Conix-security :

1. https://github.com/conix-security/machoke
2. https://github.com/conix-security/BTG

Q/A

—

Robin Marsollier
robin.marsollier@conix.fr
CERT-Conix