ROPMEMU:
A Framework for the Analysis of Complex Code-Reuse Attacks

Mariano Graziano, Davide Balzarotti, Alain Zidouemba

Cisco Systems, Inc.
Eurecom

AsiaCCS 2016 - Xi'an, China

TALOS

CODE INJECTIONS

TALOS

CODE INJECTIONS

OS KERNEL

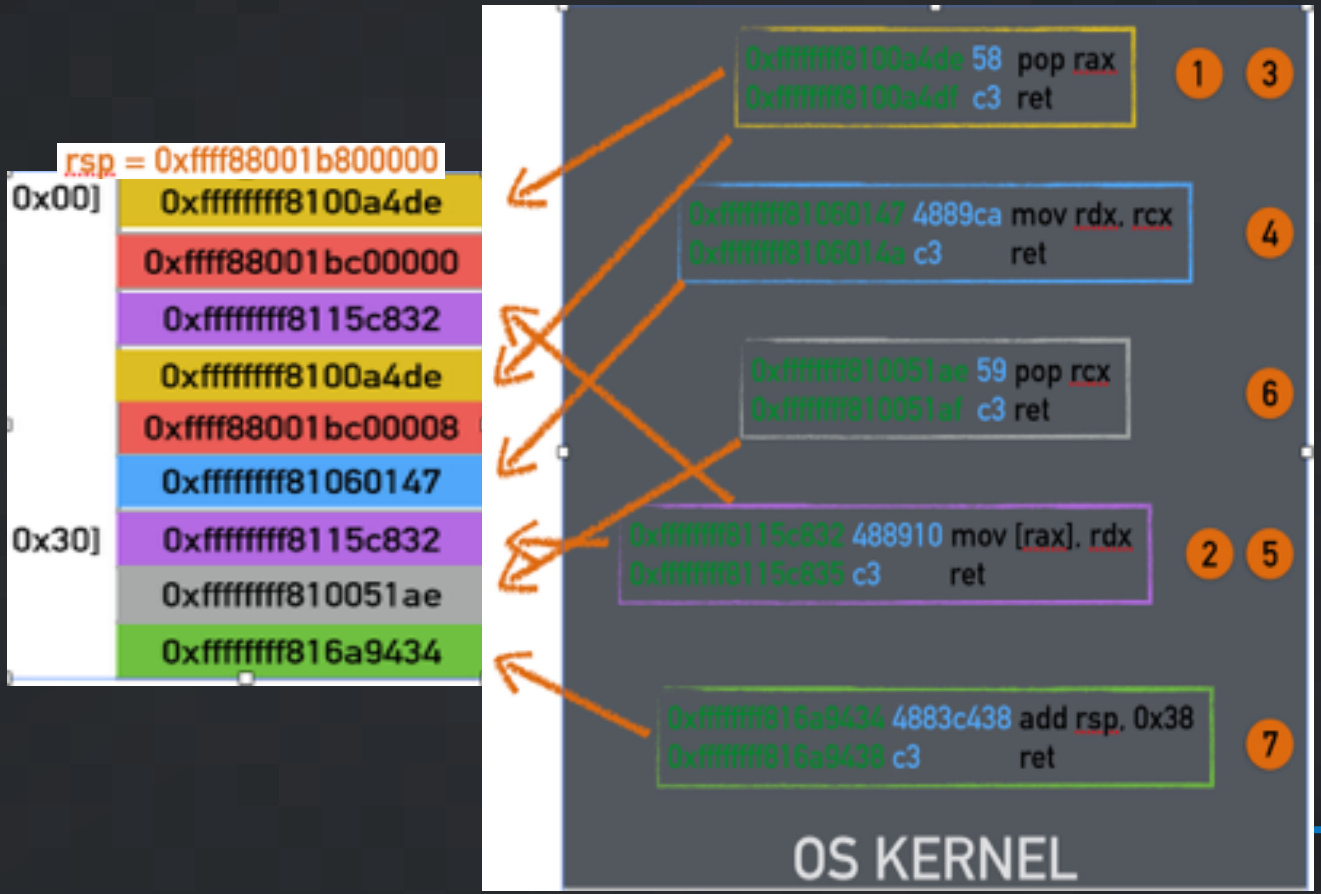
MALICIOUS CODE

```
0xc0c4c0c4 ?? ?? shellcode ins1   2
0xc0c4c0c8 ?? ?? shellcode ins2   3
0xc0c4c0d0 ?? ?? shellcode insN   N

c1000000 8b 0d c0 ae 80 01 mov ecx,ds:0x180aec0
c1000006 f6 86 11 02 00 00 40 test [esi+0x211],0x40

XYZ e9 ?? jmp 0x804b000           1
```

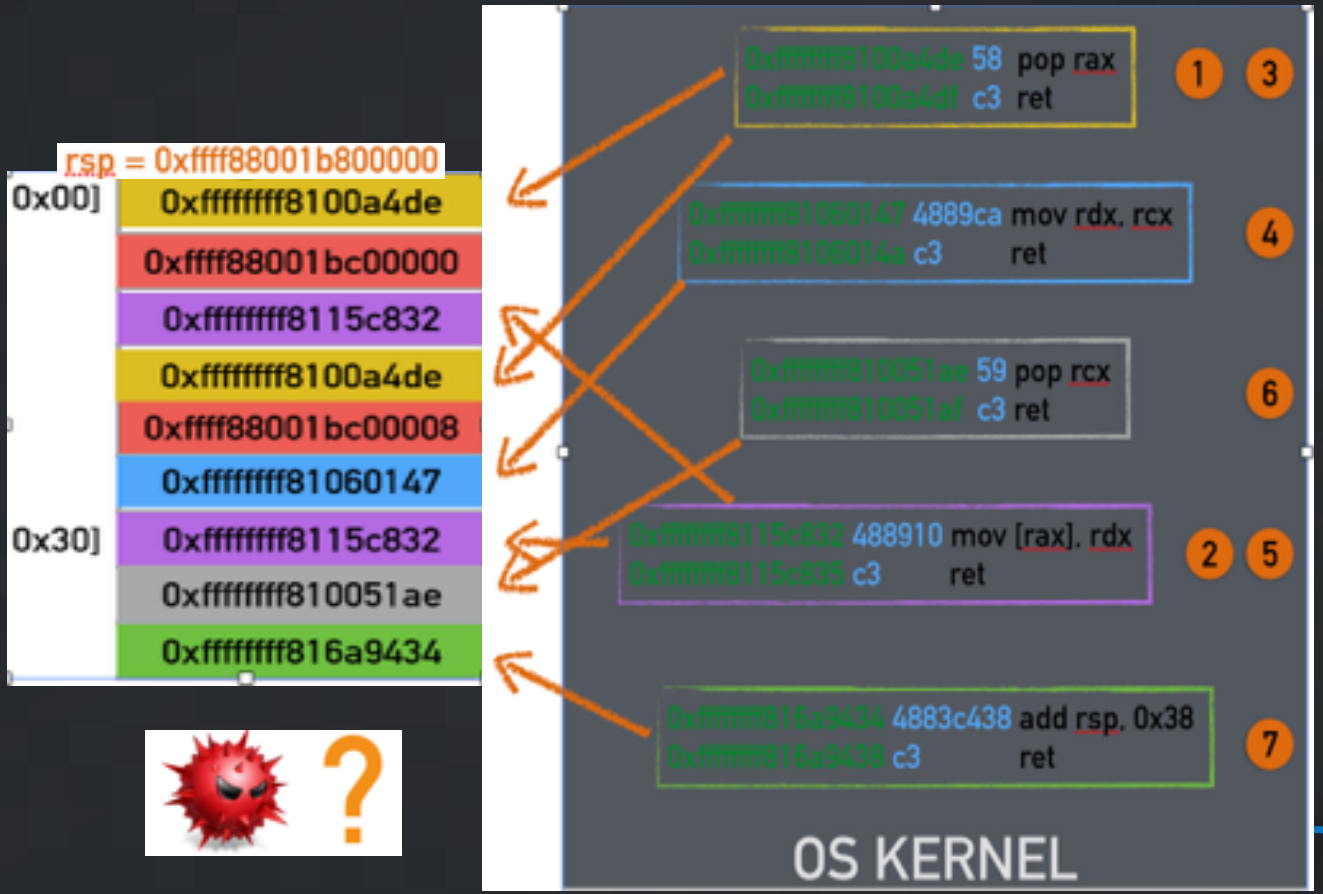
Attackers load or inject malicious code (or modify the existing one)

TALOS

CODE REUSE - ROP



CODE REUSE - ROP



TALOS

MOTIVATIONS

- HW and OS countermeasures force ROP adoption

MOTIVATIONS

- HW and OS countermeasures force ROP adoption

- Vogl et al. [NDSS 2014] — Persistent ROP rootkit
- ROP as an obfuscation technique adopted by malware
- All existing tools cope with injected code
- Lack of RE tools to analyze/dissect/decompile ROP

TALOS

CHALLENGES

TALOS

CHALLENGES

[C1] Verbosity

TALOS

CHALLENGES

[C1] Verbosity

[C2] Lack of immediate values

TALOS

CHALLENGES

[C1] Verbosity

[C2] Lack of immediate values

[C3] Stack based instruction chaining

TALOS

CHALLENGES

[C1] Verbosity

[C2] Lack of immediate values

[C3] Stack based instruction chaining

[C4] Conditional branches

CHALLENGES

[C1] Verbosity

[C2] Lack of immediate values

[C3] Stack based instruction chaining

[C4] Conditional branches

[C5] Return to functions

TALOS

CHALLENGES

[C1] Verbosity
[C2] Lack of immediate values
[C3] Stack based instruction chaining
[C4] Conditional branches
[C5] Return to functions
[C6] Dynamically generated chains

TALOS

CHALLENGES

[C1] Verbosity
[C2] Lack of immediate values
[C3] Stack based instruction chaining
[C4] Conditional branches
[C5] Return to functions
[C6] Dynamically generated chains
[C7] Stop condition

TALOS

CHALLENGES

[C1] Verbosity 

[C2] Lack of immediate values 

[C3] Stack based instruction chaining 

[C4] Conditional branches 

[C5] Return to functions 

[C6] Dynamically generated chains 

[C7] Stop condition 

TALOS

CHALLENGES

[C1]
- Lu et al. - ACSAC 2012
- Yadegari et al. - S&P 2015

[C2]

[C3]
- Stancill et al. - RAID 2013
- Lu et al. - ACSAC 2012

[C4] Conditional branches

[C5] Return to functions

[C6] Dynamically generated chains

[C7] Stop condition

TALOS

APPROACH

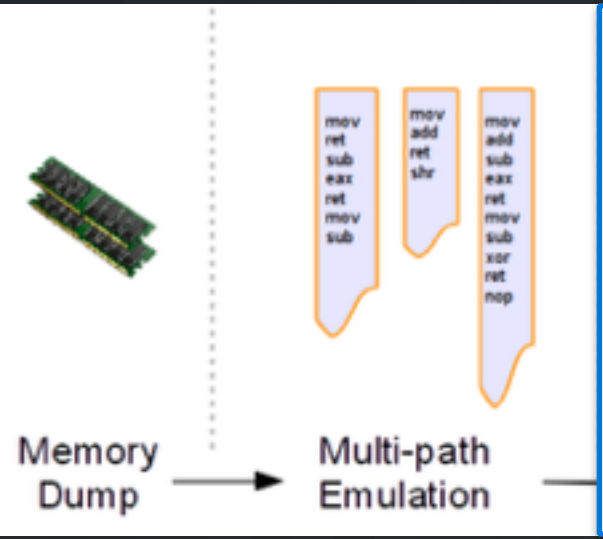
- ROPMEMU framework adopts many techniques:
  - Memory forensics
  - Code emulation
  - Multi-path execution
  - CFG recovery
  - Compiler transformations

TALOS

ROPMEMU

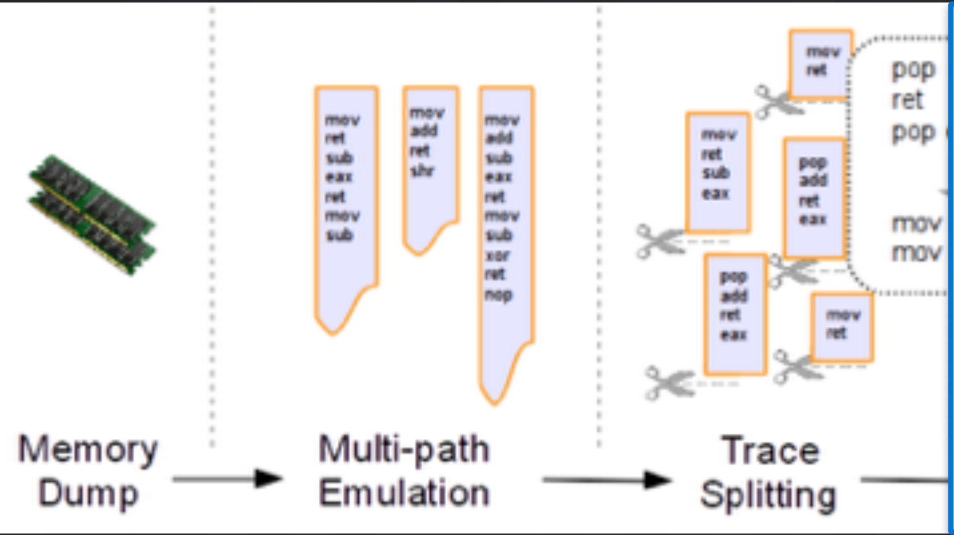
Memory Dump

ROPMEMU

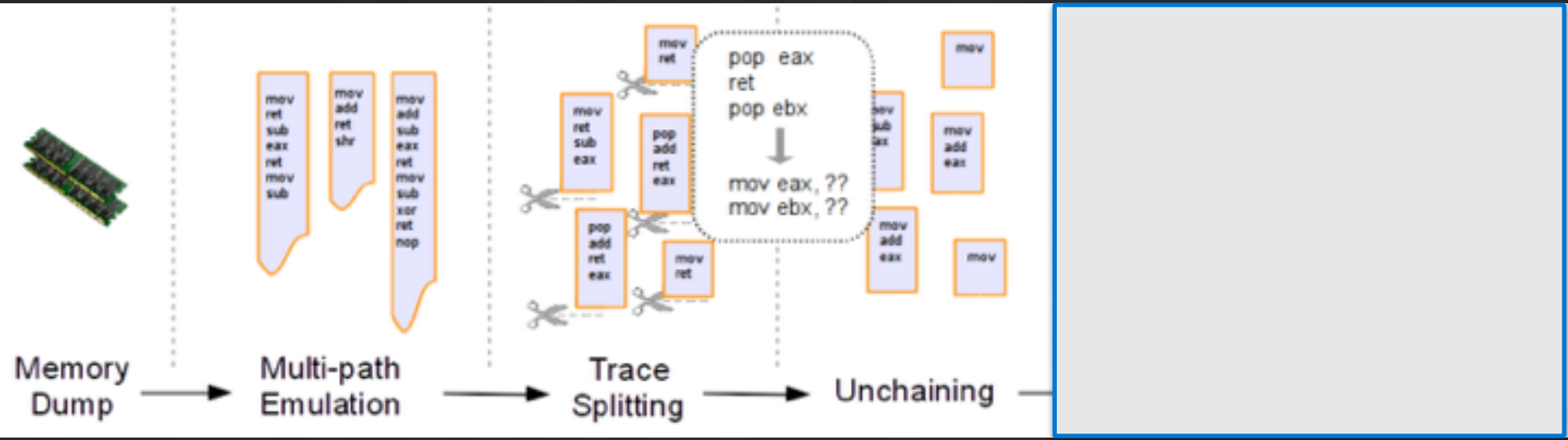


TALOS

ROPMEMU

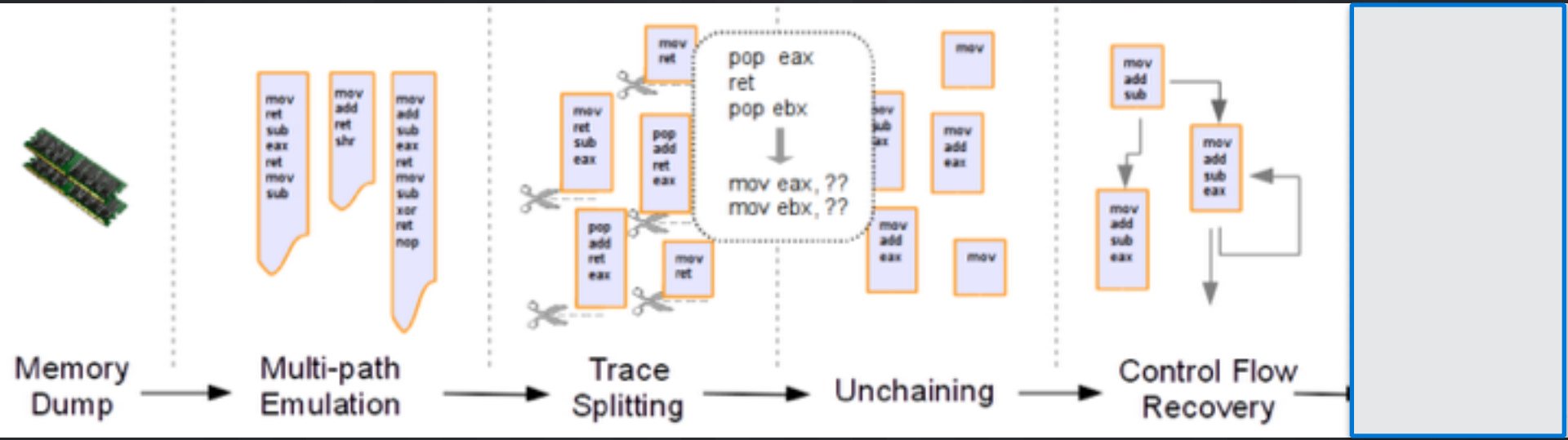

pop
ret
pop
mov
mov
Memory Dump
Multi-path Emulation
Trace Splitting

ROPMEMU



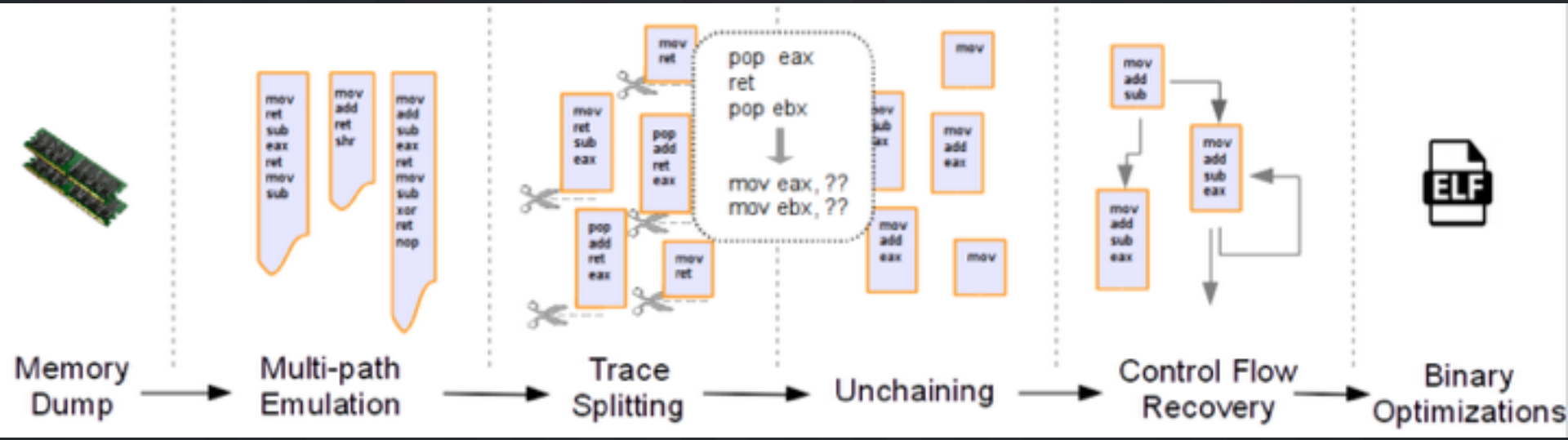
TALOS

ROPMEMU



TALOS

ROPMEMU



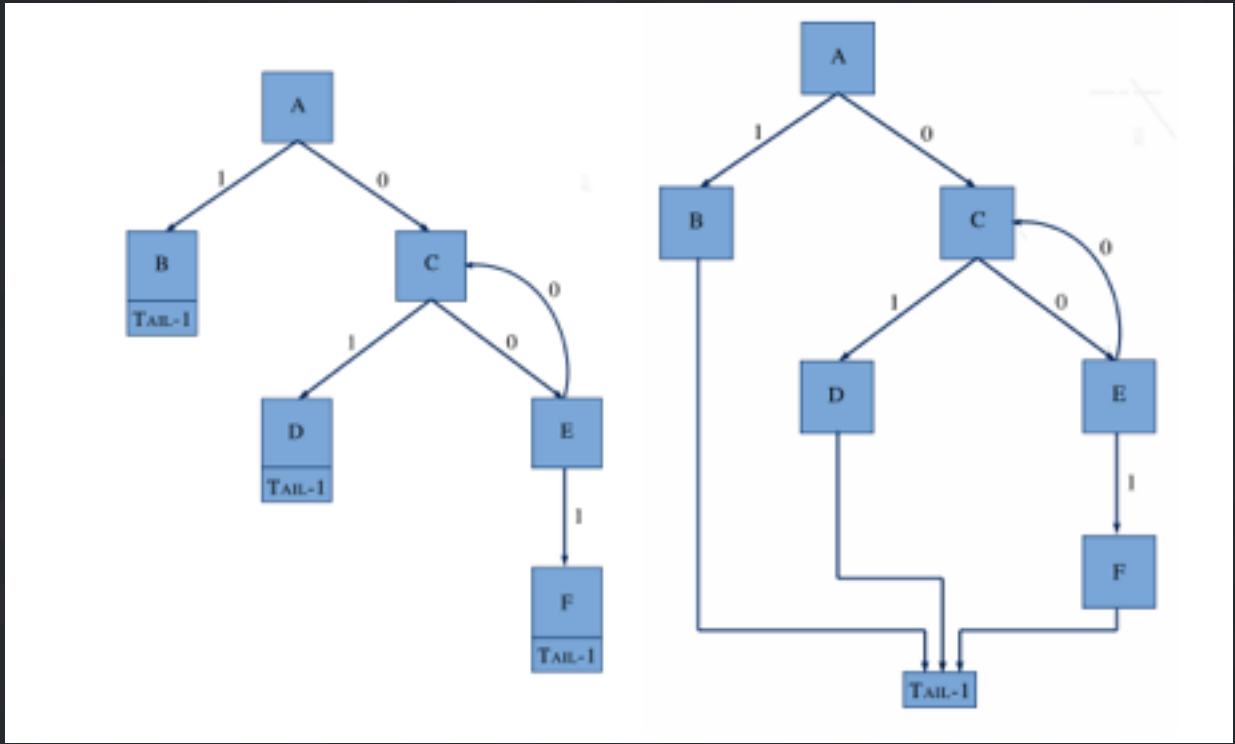
TALOS

EXPERIMENT

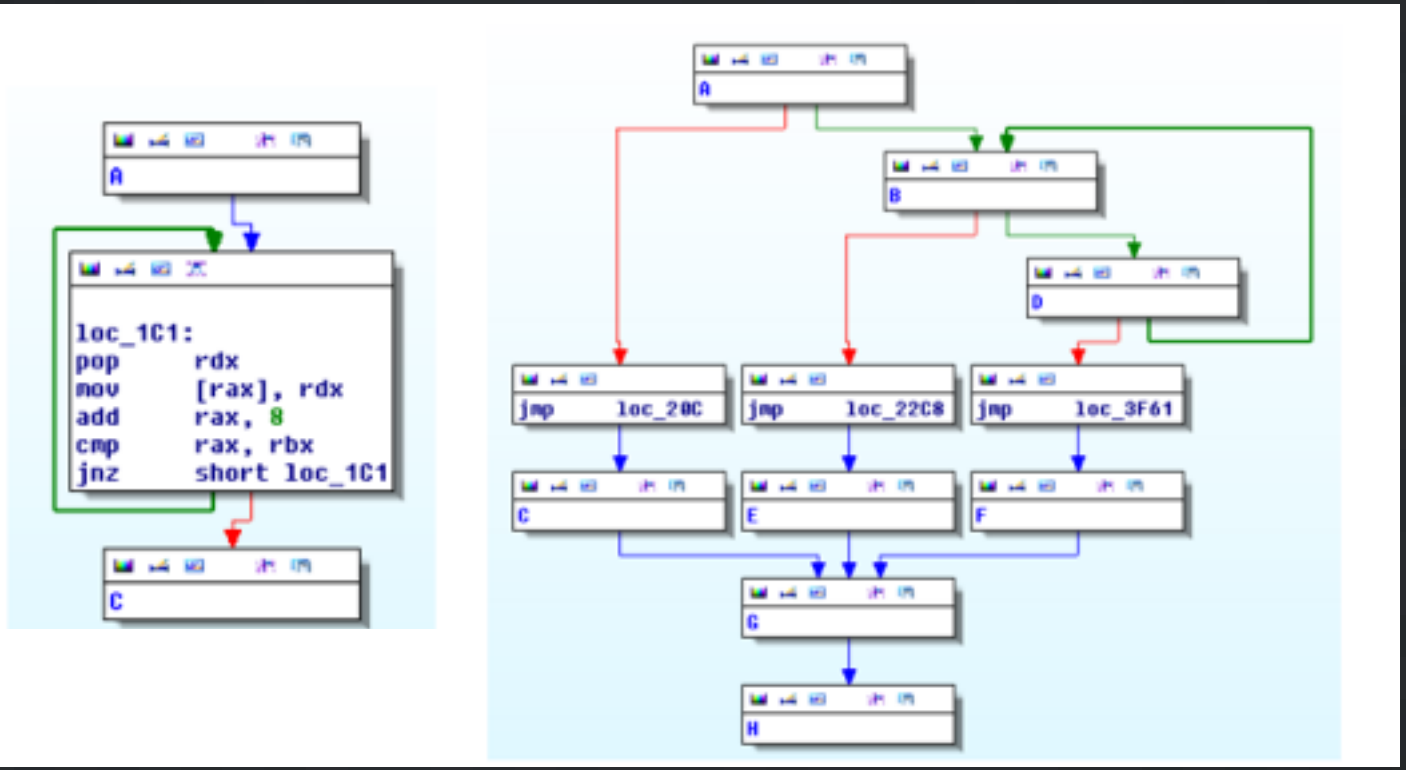
CHAIN	INSTRUCTIONS	GADGETS	BLOCKS	BRANCHES	FUNCTIONS	CALLS
COPY	414,275	184,126	1	-	-	-
DISPATCHER	63,515	28,874	7	3	1	5
PAYLOAD	6320	2913	34	26	9	17

TALOS

RESULTS - CFG

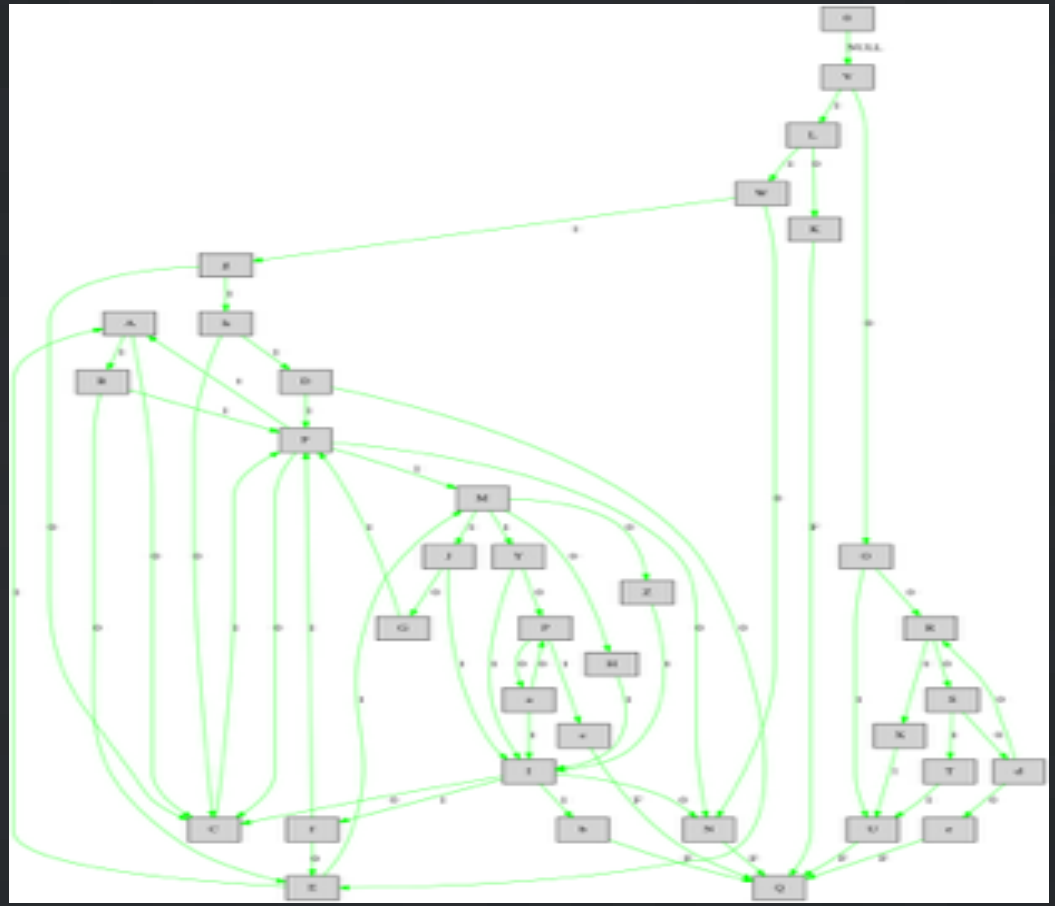


RESULTS - CFG



TALOS

RESULTS - CFG



TALOS

LIMITATIONS

- Prone to anti-acquisition
- Prone to anti-emulation
- Lack of completeness on arbitrary inputs

CONCLUSIONS

- Source code: https://github.com/vrtadmin/ROPMEMU
- First public tool to analyze ROP payloads
- Tested on the most complex public threat

TALOS

QUESTIONS?

Mariano Graziano
magrazia@cisco.com
@emd3l