Subverting Operating System Properties through Evolutionary DKOM Attacks

Mariano Graziano, Lorenzo Flore, Andrea Lanzi, Davide Balzarotti

Cisco Systems, Inc.
Università’ degli Studi di Milano
Eurecom

DIMVA 2016 - San Sebastian, Spain
TRADITIONAL DKOM ATTACKS
TRADITIONAL DKOM ATTACKS
TRADITIONAL DKOM DEFENSES

- Kernel data integrity solutions:
  - invariants
    - external systems
    - memory analysis
  - data partitioning
EVOLUTIONARY DKOM ATTACKS

data structure of interest
EVOLUTIONARY DKOM ATTACKS

Violation of a temporal property
EVOLUTIONARY DKOM ATTACKS

Violation of a temporal property.

The attack cannot be detected looking at a single snapshot.
STATE VS PROPERTY

- Traditional DKOM affects the **state** and are **discrete**

- Evolutionary DKOM (E-DKOM) affects the evolution in time of a given **property** and are **continuous**
THREAT MODEL

- Attacker has access to ring0
- Malicious code not detectable by current solutions
- Attacker cannot modify kernel code and attack the VMM
EXAMPLE: LINUX CFS SCHEDULER

```c
struct task_struct {
    volatile long state;
    void *stack;
    unsigned int flags;
    int prio, static_prio, normal_prio;
    const struct sched_class *sched_class;
    struct sched_entity se;
    ...
}

struct sched_entity {
    struct load_weight load;
    struct rb_node run_node;
    struct list_head group_node;
    ...
}

struct cfs_rq {
    ...
    struct rb_root tasks_timeline;
    ...
}

struct rb_node{
    unsigned long rb_parent_color;
    struct rb_node *rb_right;
    struct rb_node *rb_left;
};
```
SUBVERTING THE SCHEDULER

```c
struct task_struct {
    volatile long state;
    void *stack;
    unsigned int flags;
    int prio, static_prio, normal_prio;
    const struct sched_class *sched_class;
    struct sched_entity se;
};

struct sched_entity {
    struct load_weight load;
    struct rb_node run_node;
    struct list_head group_node;
    ...
}

struct cfs_rq {
    ...
    struct rb_root tasks_timeline;
    ...
};

struct rb_node{
    unsigned long rb_parent_color;
    struct rb_node *rb_right;
    struct rb_node *rb_left;
};
```

target
SUBVERTING THE SCHEDULER

```c
struct task_struct {
    volatile long state;
    void *stack;
    unsigned int flags;
    int prio, static_prio, normal_prio;
    const struct sched_class *sched_class;
    struct sched_entity se;
    ...
}

struct cfs_rq {
    ...
    struct rb_root tasks_timeline;
    ...
};

struct sched_entity {
    struct load_weight load;
    struct rb_node run_node;
    struct list_head group_node;
    ...
}

struct rb_node{
    unsigned long rb_parent_color;
    struct rb_node *rb_right;
    struct rb_node *rb_left;
};
```

*target* right most
SUBVERTING THE SCHEDULER

```c
struct task_struct {
    volatile long state;
    void *stack;
    unsigned int flags;
    int prio, static_prio, normal_prio;
    const struct sched_class *sched_class;
    struct sched_entity se;
    ...
}

struct sched_entity {
    struct load_weight load;
    struct rb_node run_node;
    struct list_head group_node;
    ...
}

struct cfs_rq {
    ...
    struct rb_root tasks_timeline;
    ...
};
```

Set \( target_{vruntime} > rightmost_{vruntime} \)
We affected the evolution of the data structure over time. We altered the scheduler property (fair execution).
ATTACK EVALUATION

- Temporarily block an IDS or Antivirus
- Temporarily block Inotify
DEFENSES?

- Reference monitor that mimics the OS property:
  - OS specific
  - Difficult to generalize
DEFENSE FRAMEWORK

CFSrb-tree

27

19

7

25

19

34

31

99

CFS-ATTACK

Raise target virtual runtime

Operating system

Hypervisor

VMExit on scheduler events

Periodic VMExit

Periodic Monitor

Inferred Task List

Update tasks info

Task Tracer

ALERT!
DEFENSE FRAMEWORK

- CFS rb-tree
- CFS ATTACK
- Raise target virtual runtime
- Operating system
- Hypervisor
- VM exit on scheduler events
- Periodic VM exit
- Periodic Monitor
- Inferred Task List
- ALERT!
DEFENSE FRAMEWORK

CFS rb-tree

27

19

7

25

31

34

99

CFSAttack

Raise target virtual runtime

Operating system

Hypervisor

VMExit on scheduler events

Periodic VMExit

Periodic Monitor

Inferred Task List

Update tasks info

Alert!

Task Tracer

OS
OVERHEAD

Normal operations

Stress test
CONCLUSIONS

‣ New DKOM attack based on data structures evolution

‣ Experiment on the Linux CFS scheduler

‣ Defense solution based on hypervisor

‣ General mitigation/solution very hard
QUESTIONS?

Mariano Graziano
magrazia@cisco.com
@emd3l