THE-DRIP-DRY-CARBONITE

Panos Sakkos

GOALS

- Merge the 2 suggested tools from "FORENSICS: Loadable Kernel Modules" article, by Keith J. Jones at the MAGAZINE OF USENIX & SAGE

- Support modern versions of the linux kernel

- Incorporate modern security mechanisms

BEFORE WE START

- LKMs (Linux Kernel Modules)
  - Object files that patch the kernel dynamically (like drivers)
  - No reboot needed
  - Need root privileges to handle LKMs

BEFORE WE START

- rootkits
  - LKMs that make the kernel untrusted
  - Hide resources (themselves, processes, connections, files etc)
  - B - Z of the attacks on linux boxes
  - The attacker needs to escalate to root

JONES'S WORK

- syscall_sentry
- carbonite

SYSCALL_SENTRY

- Copies the sys_call_table when inserted into the kernel
- Checks if the sys_call_table is the same with its copy
  - Every 10 seconds
  - Every time a kernel module is inserted into the kernel

CARBONITE

- Freezes the box and takes a snapshot of the processes
  - Dumps PID, UID, status, name, start time, open files, command line arguments and environment variables
- Dumps the data into a file

FLAWS

- Both syscall_sentry and carbonite were too easy to be fooled and bypassed
- Bounded to specific linux kernel versions (2.2 and 2.4)
- Dispatched vfs calls, in order to handle the dump file
- It's a hack, not a carefully designed system

THE-DRIP-DRY-CARBONITE

- carbonite + (proper) monitoring of the sys_call_table
  - Hashes sys_call_table symbol (Super Fast Hash)
- Supports 2.6.32 - 3.4.2 (at least)
- Offers tuning of sys_call_table monitoring
- Automatic dispatch of carbonite after incident detection
- User-space tool for triggering carbonite

THE-DRIP-DRY-CARBONITE

- carbonite is exported as kernel symbol
  - in order to be utilized by other security LKMs

- Logs remotely to a syslog server in order to back up logs in case of a sophisticated attack
- Stealth mode, hides itself form the kernel and it cannot be removed
- Open source: https://github.com/PanosSakkos/the-drip-dry-carbonite

COMPLEXITY

90 loc (syscall_sentry) + 500 loc (carbonite)

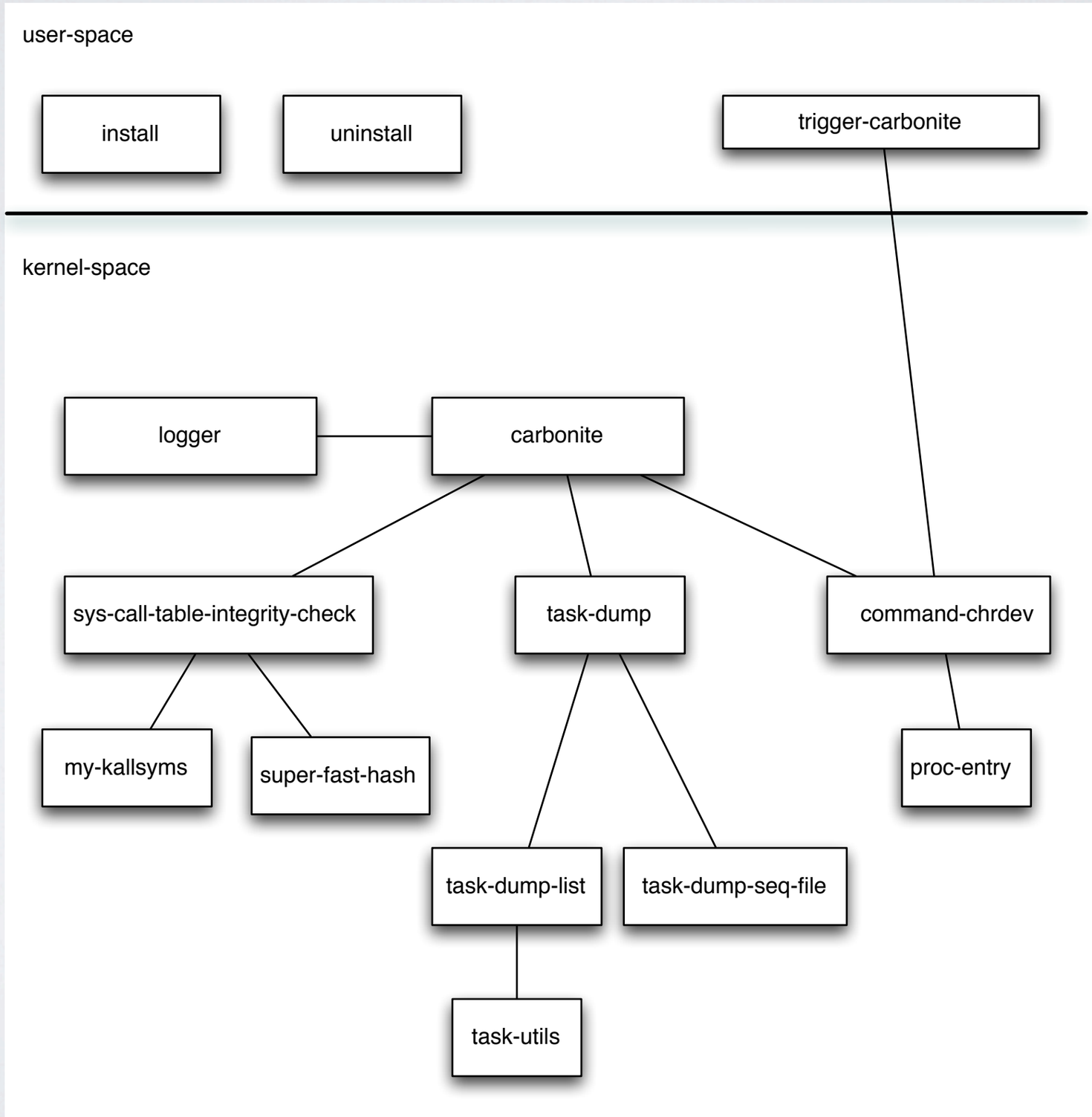
vs

1879 loc (the-drip-dry-carbonite)

```
void carbonite(int why)
{
        int ret;
        unsigned long flags;
        struct task_struct *task;
        DEFINE_SPINLOCK(carbonite_spinlock);

        if(previous_reason == AFTER_MODULE_INIT && why == AFTER_INCIDENT)
        {
                /* Don't trigger carbonite twice for an attacking module,
                 * because the snapshot of the processes will not be accurate.
                 */

                 previous_reason = why;

                 return;
        }

        print_dispatch_reason(why);

        /* Stop the world */

        spin_lock_irqsave(&carbonite_spinlock, flags);

        for_each_process(task)
        {
                ret = dump_task(task);

                if(!ret)
                {
                        print_log(KERN_ERR MODULE_NAME ":\t[-] Failed to dump %s task with PID %d\n", task->comm, task->pid)
                }
        }

        spin_unlock_irqrestore(&carbonite_spinlock, flags);

        previous_reason = why;
}
EXPORT_SYMBOL(carbonite);
```

the-drip-dry-carbonite is not a hack, it's an elegant and effective system

ARCHITECTURE



DEMO

FUTURE WORK

- Dump open files, command line arguments, environment variables and executables
- Report (explicitly) hidden processes

QUESTIONS

THANK YOU