Protecting Suspended Devices from Memory Attacks

EuroSec 2017, Apr 23, 2017

Manuel Huber, Julian Horsch, Sascha Wessel

Fraunhofer AISEC

Outline

Introduction

Protection of Linux Systems

Protection of other OSs

Performance and Security

Conclusion

Introduction
Motivation

Problem: Common systems keep main memory in plaintext

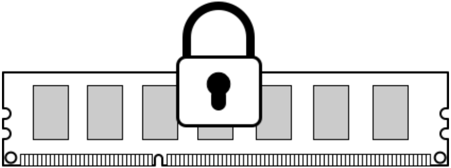
- Makes devices left unattended especially susceptible to memory attacks
- Physical attackers can retrieve main memory via JTAG, DMA, Coldboot

Valuable targets: FDE key and sensitive process memory (passwords, classified information, ...)

Goal: Protect sensitive data on stolen/unattended x86 devices from disclosure

Introduction
Contributions

Idea: Encrypt the RAM possibly containing sensitive data on suspension and decrypt on resumption



Contributions:

- Generic design of a lightweight RAM encryption kernel mechanism
- Prototype for current Linux kernel versions
- Integration into a hypervisor to transparently protect other OSs
- Performance and security evaluation

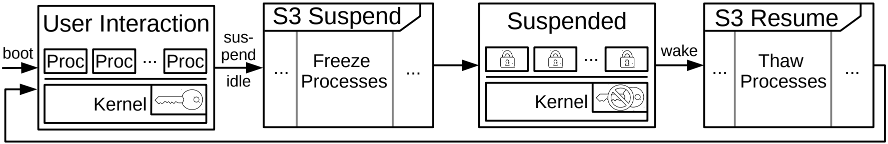
Introduction
Previous Work

- **Hardware based approaches:** XOM, Aegis, processor extensions, ...
  - Hardware extensions typically not available on consumer devices
  - Processor extensions only serve as a building block for memory protection (re-design software)
- **Key hiding:** TRESOR, Copker, ...
  - Move keys to CPU/GPU registers or caches
  - Protect specific assets only
- **Software-based approaches:** Hypercrypt, RamCrypt, Hypnoguard, ...
  - Smaller coverage leaving possibly sensitive segments uncovered
  - Sliding windows with undefined unencrypted memory
  - Adversely affect performance with runtime encryption
  - Some on ARM, some rely on specific (legacy) hardware characteristics

Protection of Linux Systems
Design Goals

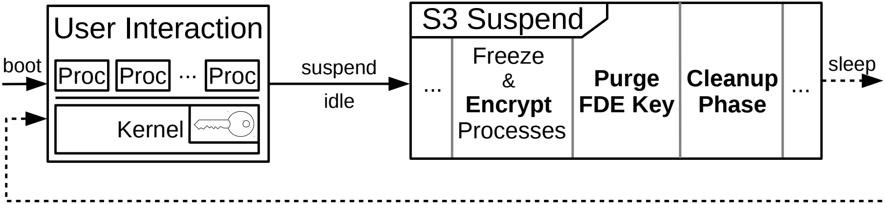
- **Lightweightness:** No configuration, ease of integration
- **Usability:** No implications on workflow, common authentication
- **Hardware Independence:** No dependencies, simple portability
- **Performance:** No lengthy resumption, seamless suspend and wakeup cycles
- **Coverage:** No confidential data left unencrypted

Protection of Linux Systems
Overview and Observations



- **Wake state:** Sensitive data in kernel and userspace while system in use
- **Desired Suspend State:** Encrypted process memory and no sensitive contents in kernel memory
- **Observations:**
  - Common systems use FDE with LUKS for storage protection
  - Processes are frozen on suspend and thawed on wakeup

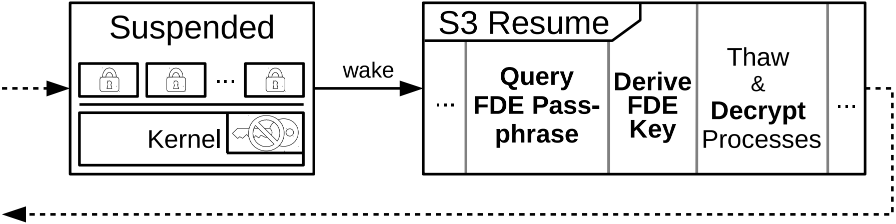
Protection of Linux Systems
Redesign of the Suspension



- Use the FDE key to encrypt confidential data in RAM
- Every freezing task encrypts its memory regions in parallel
- Remove FDE key and other sensitive kernel data

Protection of Linux Systems

Redesign of the Resumption

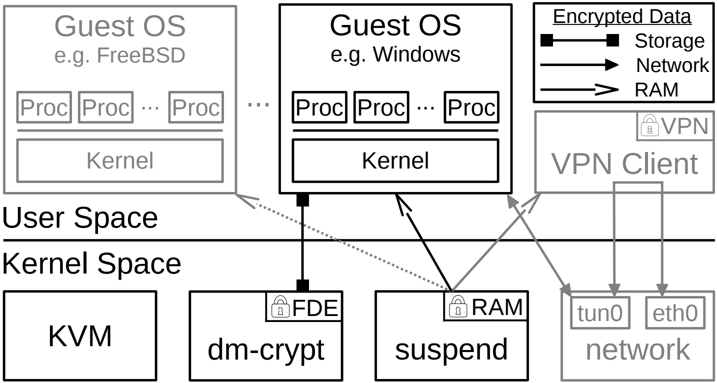


- Query passphrase from user
- Restore FDE key based on LUKS header (key derivation)
- Re-supply key to storage encryption
- Decrypt process memory before thawing processes

Protection of Linux Systems
Implementation

- **En-/decryption:**
  - Shared kernel structures (virtual memory regions) between threads
    $\longrightarrow$ Only last task of a process encrypts (same one decrypts)
  - Shared physical pages between processes
    $\longrightarrow$ Mark pages as encrypted (page flags) and reset flags on decryption
  - Use AES in CTR mode. Physical page addresses provide unique IVs for all blocks
  - Excluded: special segments (memory-mapped IO, DMA memory), shared library code segments
- **Cleanup Phase:** zero FDE key, utilized cipher structures, kernel stack regions, freed pages (list in page allocator)
- **Passphrase Query:** simple query in kernel
  $\longrightarrow$ fully operational kernel to reuse drivers and interact with devices

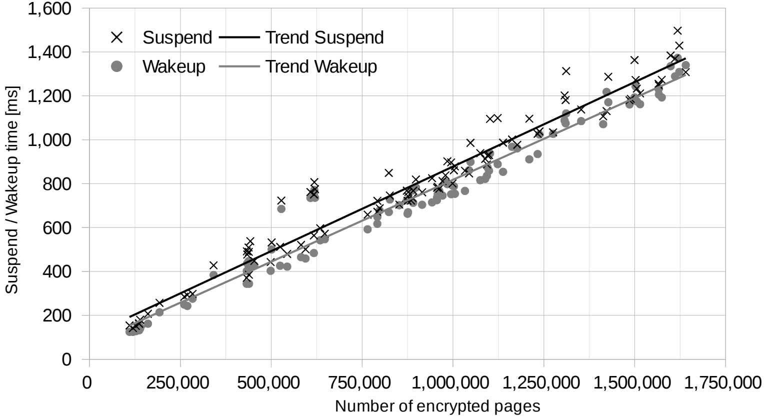
Protection of other OSs



- Use the mechanism in a hypervisor, which only appears to user on boot and resume
- Protects the full guest OS's memory
- Transparent VPN network traffic routing

Fraunhofer AISEC

Performance and Security
Performance Measurements



- Measured with AES256-CTR on a dual-core 12GB notebook (2.6 GHz)
- Linear growth, decryption slightly faster

Performance and Security
Performance Measurements

Measurement	Minimum	Maximum	Average
Processes	83	127	**105**
VMAs Encrypted	11,899	28,245	**20,379**
VMAs Total	15,241	35,050	**25,548**
Pages Encrypted	110,148	**1,640,439**	898,384
Suspend Time [ms]	142	**1,497**	807
Wakeup Time [ms]	125	**1,373**	745
Enc. Speed [MB/s]	2,860	5,106	**4,453**
Dec. Speed [MB/s]	3,081	5,387	**4,824**

Fraunhofer
AISEC

Performance and Security
Security

- **Persistent memory:**
  - Requires decryption of LUKS header
  - Can be brute-forced (PBKDF2 derivation)
  - Depends on complexity of FDE passphrase
  - TPM or SE can be easily integrated (e.g., TPM-LUKS)
- **Process memory:**
  - All confidential VMAs encrypted
  - Efforts coincide with the decryption of persistent memory
- **Kernel memory:** removed FDE key, cleaned memory
  → complete cleanup for productive devices required (e.g., IPSec key)
- **Freed memory:** zero'd out

Conclusion

- Devices are protected from memory disclosure once suspended
- Fulfilled design goals:
  - ✓ **Lightweightness:** Comes as an easy to deploy kernel patch
  - ✓ **Usability:** Requires FDE passphrase for authentication only and allows to disable screen lock
  - ✓ **Hardware Independence:** Works straightforward on Linux systems
  - ✓ **Performance:** On a common device, wakeup delay of less than 1.4 seconds
  - ✓ **Completeness:** Encrypts, resp. purges, all sensitive data

Conclusion
Thank you for your attention!

Questions, Answers, Discussion, ...

Conclusion
Comparison with Hypnoguard

Hypnoguard (presented at CCS 2016):

- RAM encryption on suspend/wakeup cycles using a TPM and Intel TXT
- Hook into stages where kernel is inactive
- Encrypts (almost) all RAM in one chunk including kernel memory
- Missing support of the OS and drivers:
  - Complicates portability (displays, keyboards)
  - Makes implementation of hardware-specific drivers and crypto routines necessary

Contact Information

Manuel Huber, Julian Horsch, Sascha Wessel

Secure Operating Systems

Fraunhofer-Institute for
Applied and Integrated Security (AISEC)

Address: Parkring 4
85748 Garching (near Munich)
Germany
Internet: http://www.aisec.fraunhofer.de

Phone: +49 89 3229986-165
Fax: +49 89 3229986-222
E-Mail: manuel.huber@aisec.fraunhofer.de

Fraunhofer
AISEC