

# A Linux Kernel Auditing Tool for Host-Based Intrusion Detection

William A. Maniatty, Adnan Baykal, Vikas Aggarwal,  
Joshua Brooks, Aleksandr Krymer and Samuel Maura

baykal@cs.albany.edu

# Overview

## Security management approaches

- Prevention - slows down attackers
- Detection - alert administrators when attacks occur
- Containment - limit damage of a successful attack
- Recovery - repair (if feasible) the damage

## Where to apply these approaches

- Host based
- Network based

## Our focus on host-based intrusion detection.

- We needed to generate larger amounts of high quality audit data
- This is a lot like debugging, so extend a debugging tool
  - ▷ We chose Yaghmour and Deganais' Linx Trace Toolkit (LTT) [7]
    - ▷ Need to observe all system calls
    - ▷ Need to monitor all security credentials of processes

# Instrumentation Design Choices

We instrumented each system call.

We did NOT use wrapper functions for systems calls

- Many tools do this (easy to code, e.g. systrace [4]).
- But tends to cause conflicts when using more than one of these tools.

Used conditional compilation and inlining to minimize overhead

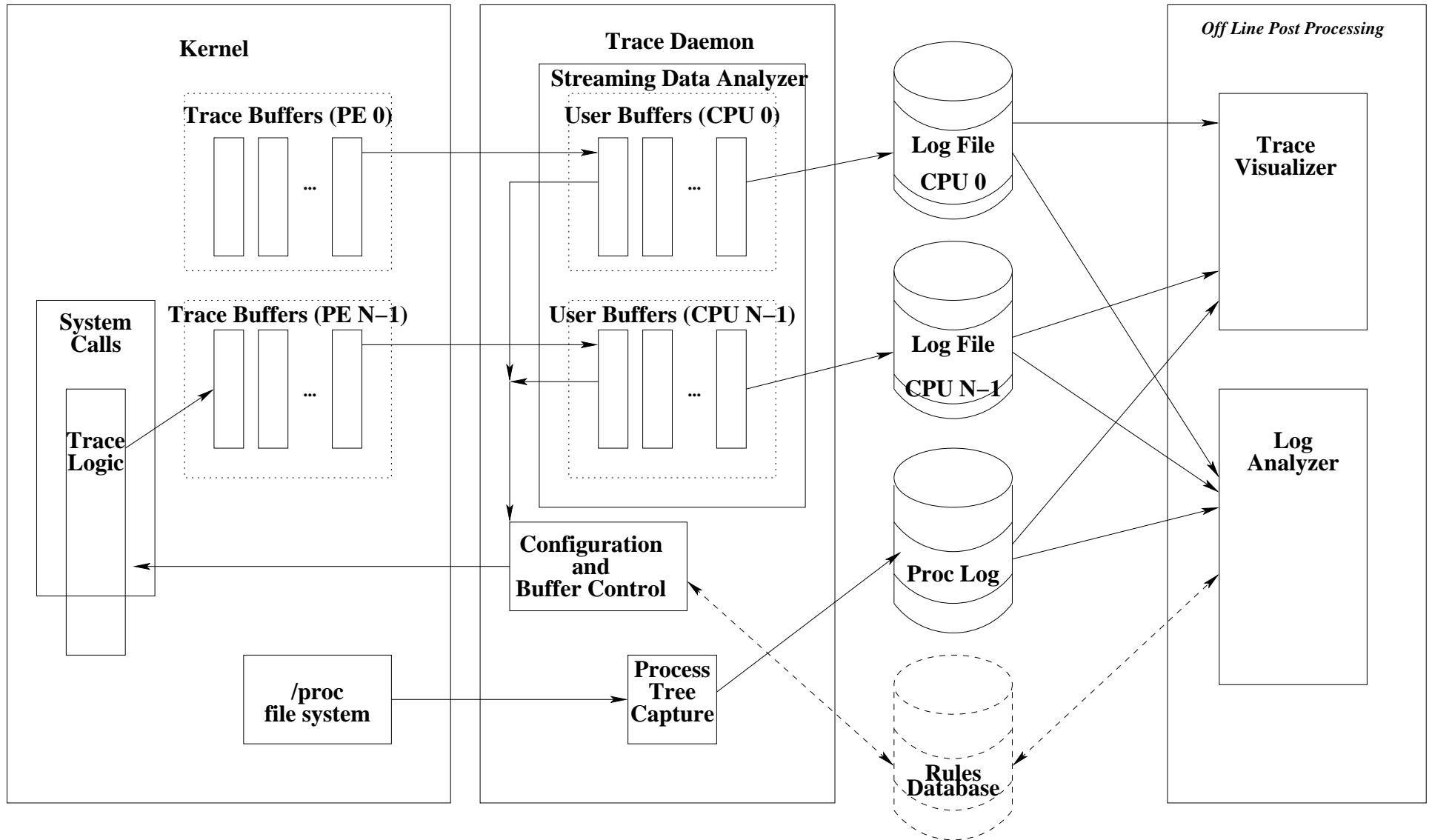
Has Useful Related Projects/Features

- RelayFS [6] (we use this)
- Flight Recorder
- DProbes [3]
- Kernel Hooks [2]

Allows for native mode tracing

- Allows for tracing on SMP systems (unlike ReVirt [5])
- Eventually, we want optional support for virtualization and replay.
- Thus, we are considering Xen [1].

# Architecture Used



# Data Management issues

We get a lot of data

- Tracing all system calls on a lightly loaded system gives about 40 GB/24 hours.
- We use inline compression (zlib), and get around 95 % size reduction
- Observed run-time slowdown of only 3-5% on I/O intensive tasks.

Support misuse detection (e.g. privilege escalation)

We are looking into mining techniques for our data

- Time series based (currently we are trying wavelet based approaches).
- Sequence based.
- Clustering and classification (principle component analysis).

Challenges of mining

- The data is highly dimensional and categorical
- This makes clustering hard.
- Use relational DB (PostgreSQL) to manage a condensed data representation
- Need streaming approaches (first get good accuracy, then stream).

## Instrumentation details

A good audit can log all inputs and results of system calls.

- Including Security credentials

```
uid_t  uid , euid , suid , fsuid ;
gid_t  gid , egid , sgid , fsgid ;
int    ngroups ;
gid_t   groups [NGROUPS] ;
kernel_cap_t  cap_effective , cap_inheritable , cap_permitted ;
int    keep_capabilities : 1 ;
struct user_struct *user ;
/* Other useful security related fields */
int    did_exec : 1 ;
void  *security ;
```

# Concluding Remarks

- Policy instrumentation needs different from auditing
  - ▷ E.g. Linux Security Modules has user exit style programming and lkms.
  - ▷ Hooks placed before major changes to kernel state (does not record results).
  - ▷ Sometimes not invoked (i.e. if parameters fail sanity checks).
  - ▷ Only “security related” calls have instrumentation so exploitable calls (e.g. mmap) may lack hooks
  - ▷ We are investigating combining LSM (policy) and LTT (audit)
- Virtual Machines provide containment and tamper resistance
  - ▷ But many users run native mode, and need logging
  - ▷ Ideally a useful tool should work in BOTH virtual and native mode
- Offers strong misuse detection and response
  - ▷ Highly detailed flight recorder dumps
- Close coupling of mining and acquisition looks promising
  - ▷ Begin with traditional clustering, time series and sequence mining
  - ▷ Then pick streaming approaches for improved accuracy
- Responses and policy integration a work in progress

# Bibliography

## References

- [1] B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, I. Pratt, A. Warfield, P. Barham, and R. Neugebauer. Xen and the art of virtualization. In Proceedings of the ACM Symposium on Operating Systems Principles, October 2003.
- [2] R. J. Moore. Generalised kernel hooks interface - a high speed call-back mechanism for the linux kernel. In Proceedings of the UKUUG Linux 2001 Linux Developers' Conference, Manchester, England, July 2001. On line at <http://www-124.ibm.com/linux/projects/kernelhooks/>.
- [3] R. J. Moore. A universal dynamic trace for linux and other operating systems. In Proceedings of the FREENIX Track: 2001 USENIX Annual Technical Conference, Boston, Massachusetts, USA, June 2001.
- [4] Niels Provos. Improving host security with system call policies. In Proceedings of the 12th USENIX Security Symposium, Washington, DC, August 2003.

- [5] Peter M. Chen Samuel T. King. Backtracking intrusions. In Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP 2003), pages 223–236, Bolton Landing, NY USA, October 2003.
  
- [6] K. Yaghmour. RelayFS: An efficient unified approach for transmitting data from kernel to user space. In Proceedings of the Ottawa Linux Symposium 2003, Ottawa, Ontario, Canada, July 2003. The Linux Symposium’s 2003 web page is at <http://www.linuxsymposium.org/2003/>.
  
- [7] K. Yaghmour and M. R. Dagenais. Measuring and characterizing system behavior using kernel-level event logging. In Proc. of the USENIX Annual 2000 Technical Conference, pages 13–26. USENIX, June 2000.