April 20, 2000

Advisor: Mikhail J. Atallah
Chairman: Feick

Reliably Exploiting Audit Logs

Portions of this work were supported by the National Foundation, by contracts MDA904-96-1-0116 and MDA904-97-0176 from the Maryland Procurement Office, and by sponsors of CERIAS.
by others. But log tampering touches only part of the problem. So logs must obviously be protected from alteration, a problem already studied.

extracted is not accurate, or if unexpected log content can cause them to fail, and may even add vulnerabilities of their own. If the information extraction tools that exploit logs may not deliver the information from logs. Many techniques for information assurance and security involve extracting...

Motivation
Log integrity techniques (studied by others) assure only that $R = R'$.

Reliable producers and consumers that *demonstrably correspond* (addressed in this work) are also needed for assurance that $E = E'$. 
This approach was demonstrated by building a grammar for Solaris BSM logs.

checked for such problems.
For distinct events, semantic loss is inevitable. A grammar can be machine implemented to be produced.
If the specification allows ambiguous or indistinguishable records to be produced,

for semantics as well as syntax.
– A natural form is a grammar, arranged and annotated to serve as a reference.
– Formal that both producer and consumer can be shown to implement.

Document Log syntax and semantical carefully in a specification sufficiently.

observed by the Log producer).

concludes from a Log entry correctly reflects the event and system state.
The consumer to fail on reading (and semantic accuracy) what the consumer establish robustness (nothing the producer can write in the Log will cause Assured correspondence between Log producers and consumers is needed to

Approach
The last point was demonstrated as we formalized a BSM specification. Human recognition of weaknesses, ambiguities, omissions, and content.

The process of formalizing a Log specification facilitates both automatic and focus.

Concurrent to discuss, detailed specifications of existing formats provide that discussions and critiques of Log content and semantics require something.

BSM examples: ocitl, rename.

English analogy: French train conductors early to strike / to strike early.

Theory: Log produced by automation \( \iff \) syntax variation reflects state.

— Syntax overlooked is semantic loss: structure carries meaning.

Claims:

an extraneous to attend to grammar and syntax?

While the community still strives to pin down Log content and semantics, is it

Relation to Content/Semantics
Skimming approach.

ASAX, IDIOIT, and USTAT are examples of ID tools supporting BSM with a

Semantic nuances are of no concern ●

That fraction can be characterized in advance and readily distinguished ●

Small fraction of log content is of interest ●

Like skimming a natural language text for key words, can work when:

Some information can be extracted without regard to grammatical structure;

Alternatives
"Skimming"

**Drawbacks:**
- Difficulty to identify and check assumptions concerning expected input
- Some actions of a parser in later processing
- Semantics carried by syntax lost, recoverable (if at all) only by duplicating
- Invalid input detected late or not at all

**Advantages:**
- Does not require specialized tools like parser generators
- Low processing cost up front
- Conceptually simple
Issues preserved in the canonical form.

- But not ID patterns or rules—these deal with native log syntax and semantic
- Can simplify porting ID engines between platforms
- Semantics not independently specified, require familiarity with native form.
  Align data on word boundaries
- Simple rearrangements of the native records (discard fields of no interest)

Canonical forms may be called shellows. The native log, but convert parts of it first to some canonical form. Their

Shallow vs. Deep Canonicalization

Common Intrusion Detection Framework (CIDF)

Illustrated by the Common Intrusion Specification Language (CISL) of the

For some applications, skimming is not suited, such as deep canonicalization

Motivation, Revised
way, CDSL would not live up to its promise.

Either shallow form (or that explicitly and expressively misinterprets the log). Either
interpreted without knowledge of the original format (CDSL degenerates to a
formats, the results will be familiar: CDSL streams that cannot be properly
It tools (such as CDF E-boxes) do only the familiar skimminig of input log
and translate the original semantics
— but that means a CDSL canonicalizer must correctly and completely discern
— CDSL is rich enough to express the semantic nuances of the native form

can be ported between platforms with similar vulnerabilities
canonicalization, not only evaluation engines but intrusion patterns themselves
independent of any native log format. In an ID system based on deep
A deep canonical form, e.g. CDSL, has semantics explicitly specified.

Deep Canonicalization
neglected in this area

Addressed feasibility/efficiency concerns that may have contributed to historical

are detailed with hyperlinks to original BSM docs for comparison

Discrepancies and ambiguities in BSM documentation identified in the process

Available in a BSM-Parsing package for quick-start BSM-based projects

Heavily annotated Grammar for BSM through Solrnis 2.6

Completed work
Process is affected by the same syntactic feature

Or else: Parsing effort not spent up front is duplicated by all ID rules or other

...can be consulted directly in later processing

Parse early and seldom: distill the semantics into an explicit internal form that

backtracking: A parser for FSM can be efficient, mostly LL(1) with some localized, bounded

Efficiency
Future directions