METAL

A tool for extracting attack manifestations

Ulf Larson, Emilie Lundin-Barse, Erland Jonsson
Computer security group
Dept. of Computer Science and Engineering
Chalmers University of Technology, Sweden

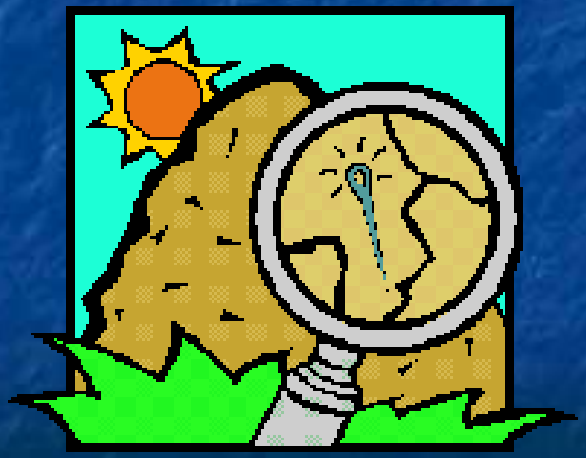
DIMVA July 7th 2005, Vienna, Austria

Presentation outline

- Introduction
- The research problem and our solution
- Manifestation extraction framework
- The METAL tool
  - Overview, components, classification, manifestation types, output data
- Results
- Conclusions

The research problem

- Q: Given a set of log data, how do we discriminate the items that were caused by an attack from benign items?



Our solution

- Lundin-Barse proposed an 8 step Framework for finding differences, or manifestations
- Manifestations were extracted by comparing logs captured during normal operation with logs captured during an attack
- Manual comparison was used

Manual comparison was used…
But… aren't logs large? And don't they contain a lot of events caused by a lot of processes?

The research problem, redefined

- Q: Given a set of log data, how do we **efficiently** discriminate the items that were caused by an attack from benign items?

Our solution, redefined

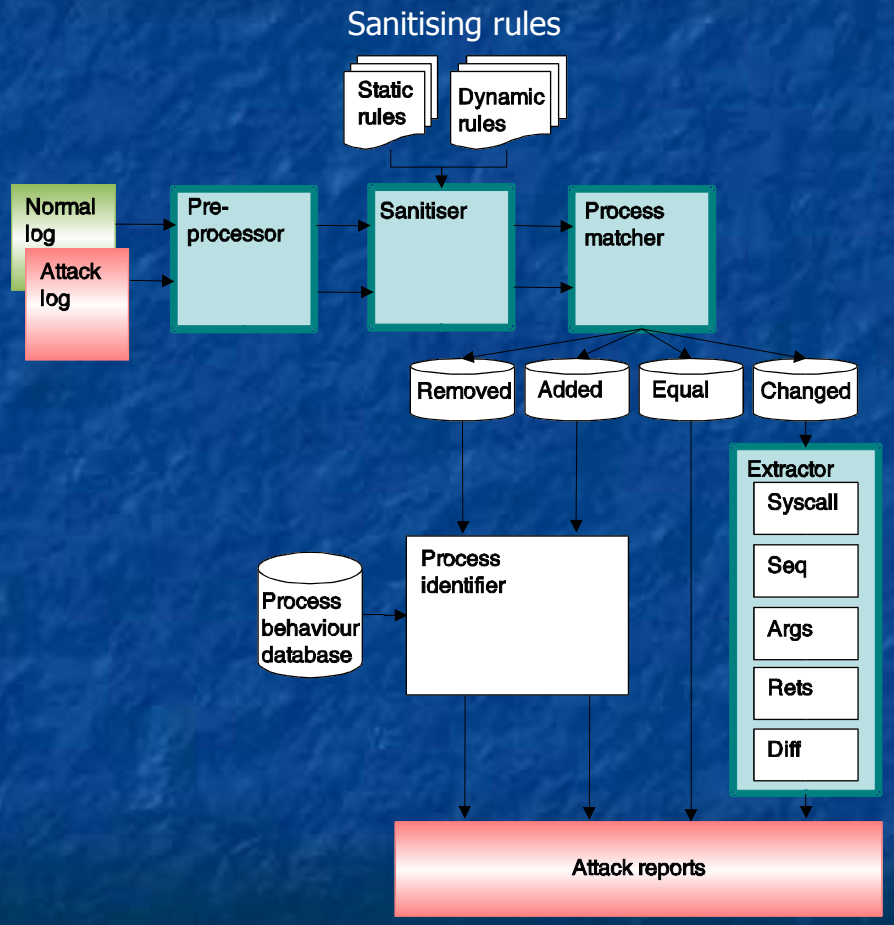
- We developed a tool, METAL, that automatically finds and extracts the differences
- We use METAL for the time consuming part of framework

Manifestation extraction

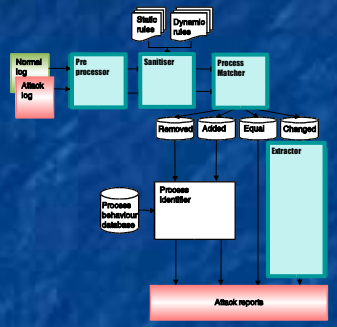
- Idea based on 8 step novel framework proposed by Lundin-Barse
  - Step 1-4: Identify attack, run attack, run corresponding normal behavior
  - Step 5: Manually compare logs to extract relevant differences
  - Step 6-8: Classifiy attacks and create log data requirements from observed differences
- METAL automates time consuming 5th step
  - Time consuming process to perform manually
  - Easy to miss or skip items due when manually analyzing the logs.
- Log source used is a system call logging tool called syscalltracker

The METAL tool: overview

- Input data
  - Normal log
  - Attack log
  - Sanitising rules
- Action components
  - Preprocessor
  - Sanitiser
  - Process matcher
  - Extractor
- Output data
  - Attack reports
  - Attack overview (relationship tree)



The METAL tool (2): components



Preprocessor

Input data:
Normal Log & Attack Log

Output data:
One file for each process in input logs divided in A and N

Sanitiser

Input data:
One file for each process
Rules for dynamic and static sanitising of the logs

Output data:
One file for each process with natural differences removed

Process matcher

Input data:
One file for each process in input logs

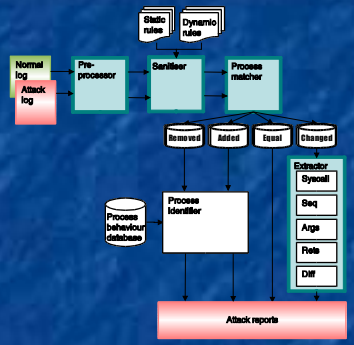
Output data:
File with score for how well processes were matched
High score: bad match, low: good

Extractor

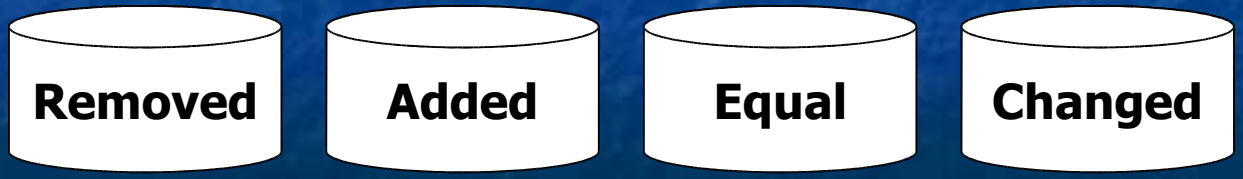
Input data:
Scorefile

Output data:
Attack reports containing differences for processes that are changed

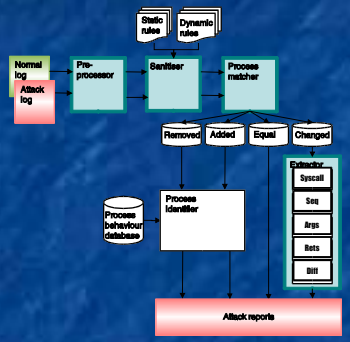
The METAL tool (3): classification



- Processes are classified depending on process matcher equality value
  - Value between 0 and 1, denotes number of sequences of certain length that matched in comp.
  - Value calculated by using percentage of equal sequences of length 6, like in "A sense for self"
- 4 classes:
  - No differences -> equal
  - Small differences -> changed
  - Large differences -> added or removed
  - Distinction between small and large depends on limit value



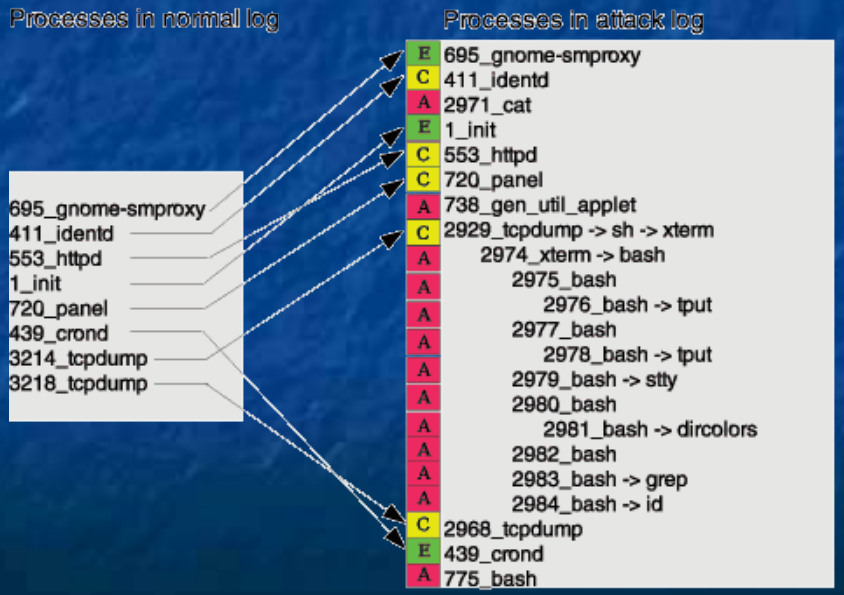
The METAL tool (4): types



- Metal extracts 5 different types of manifestations from the logs
  - Syscall: Reveals alternate program flow
    - Example: execve call to launch shell
  - Seq: Reveals alternate program flow
    - Example: adding write call before read of config file.
  - Args: Reveals use of resources, attack strings
    - Unexpected files, exploit strings
  - Rets: Reveals success of unusual operations
    - Return value of setuid or getuid calls
  - Diff: Reveals repetitions
    - Perfectly normal sequence, only repeated

The METAL tool (5): output data

- Attack overview and manifestation reports
  - The relationship between the processes are shown in the attack overview
  - For all processes that are considered as slightly changed (C), a manifestation report is created



REPORT GENERATED FOR MATCH OF SMALL CHANGES

Process from normal use of system: 3214_tcpdump
Process from attack on system: 2929_tcpdump

==========

The used sequencelength for filtering is: 6

==========

Unique system calls from [attack] 2929_tcpdump
11_execve

==========

Unique minimal foreign sequences in [attack] 2929_tcpdump

['11_execve']

==========

Unique arguments occurring in [attack] 2929_tcpdump

Syscall: 102_connect has mismatch on pos 2 for arg sockaddr{1, bffff65e}

==========

Unique diff output from running 'diff' command

> ["tcpdump"]: 11_execve("/bin/sh", CLEAN, CLEAN) (rule 11)

Results

- Manifestation extraction framework used on five attacks.
  - Three attacks previously tested manually was used as reference.
  - Comparison showed that METAL found all manifestations that were also found manually.

Attack	Type	Processes in log	Changed	Manif. examples
Tcpdump	Buffer overflow	39	5	execve + args
Wu-ftpd	Format string	39	9	Execve + args
Openssh	Privilege checking	158	48	Setuid + args
Neptune	Dos	36	8	Repeated secuence
Traceroute	Buffer overflow	39	5	-

Table 1: The results from using METAL to extract manifestations

Conclusions

- METAL significantly reduces the amount of work necessary for finding differences between log files.
- Fast and efficient identification of differences, but badly chosen reference behavior may impact matching
- The process may be useful for signature writers and security officers. Can also be used to tune a log source in order to reduce the size of logs and identify similarities between attacks.