nidsbench - A network intrusion detection system test suite

Dug Song, Gus Shaffer, Matt Undy

September 6, 1999

{dugsong,gus,mundy}@anzen.com

Abstract

Nidsbench is a lightweight, portable toolkit for testing network intrusion detection systems. It implements several well-known attacks against passive network monitoring and allows for the instrumentation of trace-driven network attack simulations.

In this talk, we describe the design and implementation of our toolkit and our experimental evaluation of several current network intrusion detection systems.

Introduction

Why test network intrusion detection systems? [RM98]

- characterize behaviour
- evaluate performance
- predict trends
- make improvements

Related Work

As lazy programmers often do, we punted at first

- UC Davis Seclab - software platform for IDS testing [UCSec96]

- IBM Zurich GSAL - IDS experimentation workbench [GSAL98]

- DARPA Intrusion Detection Evaluation - reference evaluation corpus [KK99]

- Various network traffic/attack generation tools

  – CASL, psh, tcplib, vulnerability scanners, etc.

- Various network fault injection tools

  – Orchestra, congestant, iptest, isic, etc.

Design Criteria

- Shortcomings of existing tools
  - limited event generation facilities
  - tied to specific test methodologies
  - proprietary, commercial, or platform-specific
- Design goals
  - portability
  - extensibility
  - basic software testing techniques (coverage analysis, etc.)

Implementation

Black box software testing techniques

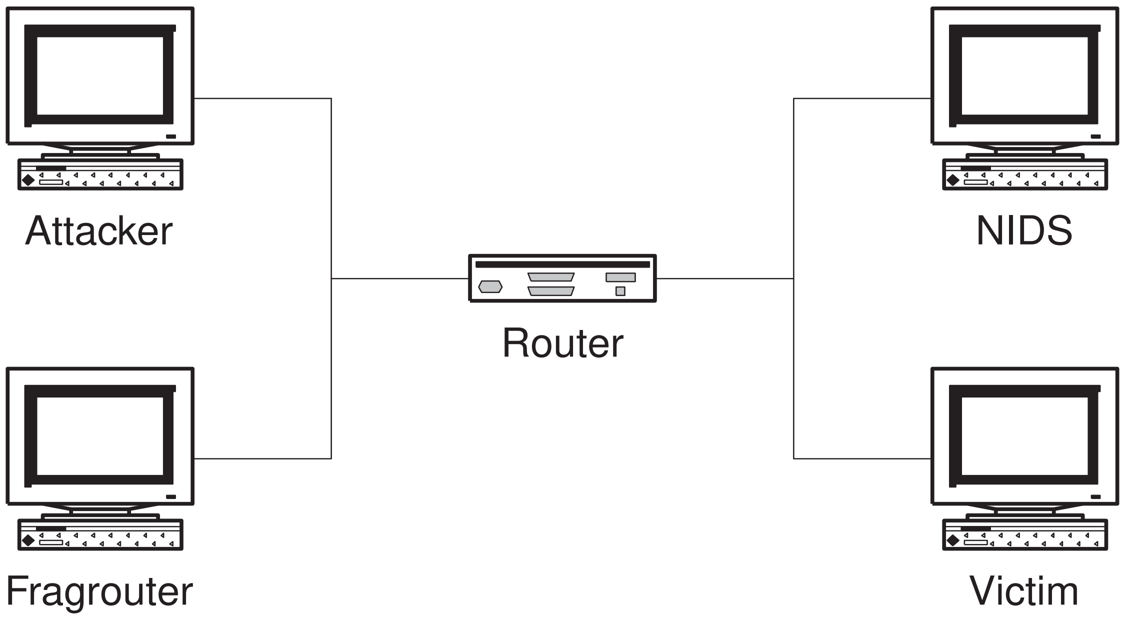
- fragrouter - fault coverage
- idstest - event coverage
- tcpreplay - trace-driven simulation

fragrouter - fault coverage

- Protocol fault injection [SJ95]
  - detect design or implementation errors
  - identify violations of protocol specifications
  - gain insight into design decisions made by the implementors
- Test specification: SNI's network monitoring bug taxonomy [PN98]
- Like Orchestra, only manipulate messages that protocol participants have generated, or generate new messages [SJM97]
- Unlike Orchestra or congestant [hor98], implemented in userland
- Based on portable open-source packet capture and generation libraries (libpcap and libnet)

fragrouter (cont.)

- Fragrouter implements an asymmetric route between the attacker and victim
  - attacker's default router is the fragrouter
  - fragrouter's default router is the router
  - victim's replies go directly to the attacker, not through the fragrouter



fragrouter tests

Name	Description	Validity
base-1	normal IP forwarding	valid
frag-1	ordered 8-byte IP fragments	valid
frag-2	ordered 24-byte IP fragments	valid
frag-3	ordered 8-byte IP fragments, one out of order	valid
frag-4	ordered 8-byte fragments, one duplicate	valid
frag-5	out of order 8-byte fragments, one duplicate	valid
frag-6	ordered 8-byte fragments, marked last fragment first	valid
frag-7	ordered 16-byte fragments, forward-overwriting	valid *
tcp-1	3-whs, bad TCP checksum FIN/RST, ordered 1-byte segments	valid
tcp-3	3-whs, ordered 1-byte segments, one duplicate	valid
tcp-4	3-whs, ordered 1-byte segments, one overwriting	valid
tcp-5	3-whs, ordered 2-byte segments, forward-overwriting	valid *
tcp-7	3-whs, ordered 1-byte segments, interleaved null segments	valid
tcp-8	3-whs, ordered 1-byte segments, one out of order	valid
tcp-9	3-whs, out of order 1-byte segments	valid
tcbc-2	3-whs, ordered 1-byte segments, interleaved SYNs	valid
tcbc-3	ordered 1-byte null segments, 3-whs, ordered 1-byte segments	valid
tcbt-1	3-whs, RST, 3-whs, ordered 1-byte segments	valid
insert-2	3-whs, ordered 1-byte segments, bad TCP checksums	invalid
insert-3	3-whs, ordered 1-byte segments, no ACK set	invalid

idstest - event coverage

- Basic hit / miss / false alarm metric
- Implement known attacks
  - publically-available vulnerability databases and exploits (CERT, BUGTRAQ, etc.)
  - vulnerability scanners often test for symptoms, instead of actually performing exploits
- Problem: lack of a standard attack or vulnerability taxonomy makes verification difficult
  - naming conventions
  - event granularity
  - attack variability

idstest (cont.)

- Introduce common attack variations, including subterfuge attacks
  - e.g. phf attack

    ```
    GET /cgi-bin/phf HTTP/1.0\r\n\r\n
    GET /cgi-bin/phf?Qalias=x%0a/bin/uname%20-a\n
    GET %2fcg%69%2d%62in%2fph%66? HTTP/1.0\n\n
    GET /c%67i-bi%6e%2fp%68f?\n
    ```

  - e.g. telnet [Pax98]

    ```
    login: root
    login: rb^Hoot
    login: ro<option>ot
    root (typeahead)
    login:
    ```

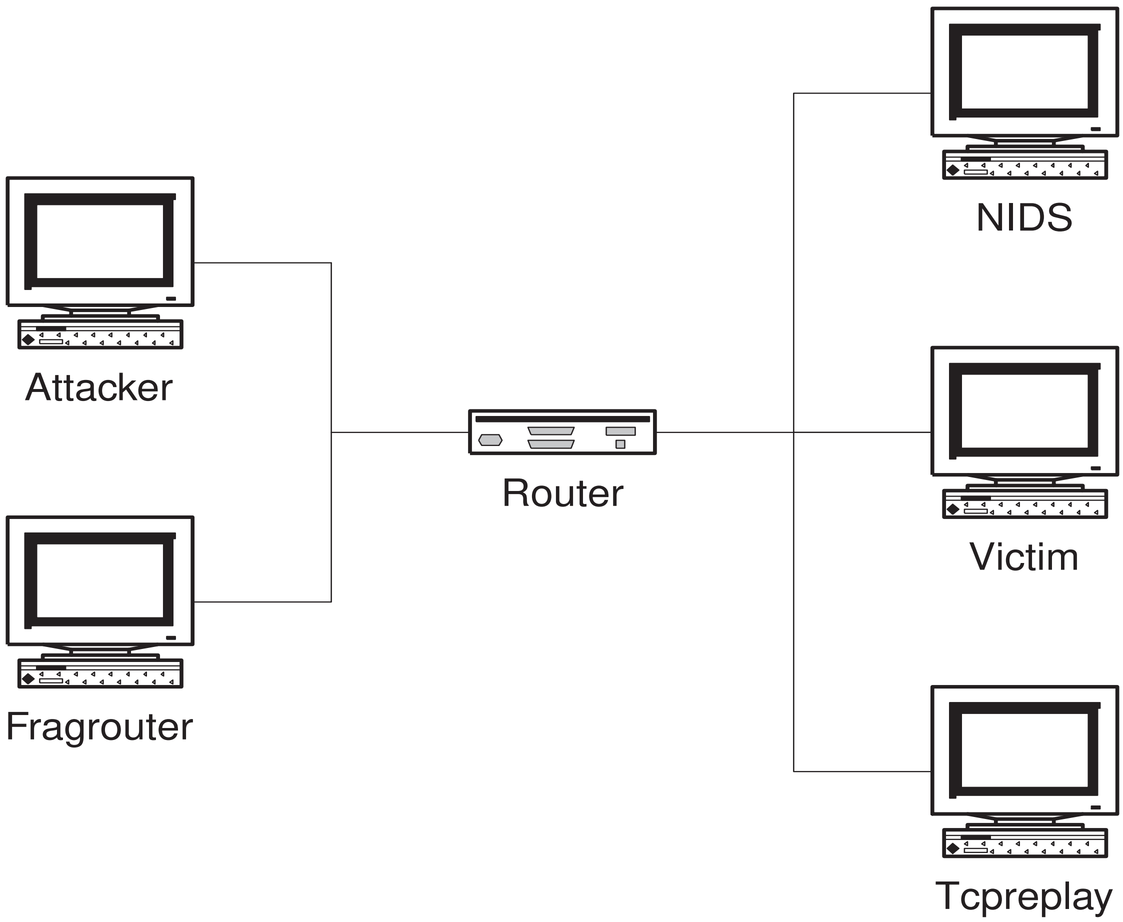
- Never publically released
- Scrapped in favor of other reference attack corpora

tcpreplay - trace-driven simulation

- Needed a way to replay saved network traffic dumps
  - e.g. 1998 DARPA IDEVAL tcpdump traces
- Support for variable-speed replay
  - flat-rate or as a multiple of the original speed, preserving std.deviation
  - this is arguably wrong because of traffic shaping [PF97], but we left it in as a poor man's load generator
- Relatively high fidelity (preservation of link-layer frames)

Experimental Environment

- Isolated fast Ethernet network split into two segmented networks, required for the fragmentation tests
- All other attacks were sent directly to (or replayed on) the IDS network



Results: fragrouter

test	valid	Bro	Dragon	NFR	RealSecure	SessionWall	NetProwler
B1	x	x	x	x	x	x	x
F1	x	x	x	x	x		
F2	x	x	x	x	x		
F3	x	x	x	x	x		
F4	x	x	x	x	x		
F5	x	x	x	x	x		
F6	x	x	x		x		
F7	x	x	x	x	x		
T1	x	x	x	x	x	x	
T3	x	x	x	x	x	x	
T4	x	x	x	x	x	x	
T5	x		x	x	x		
T7	x	x	x	x	x	x	
T8	x	x	x	x	x	x	
T9	x	x	x	x	x	x	
C2	x	x	x	x	x	x	
C3	x	x	x		x	x	
R1	x	x	x				
I2		x	x		x		
I3		x	x		x		x

Results: tcpreplay w/ IDEVAL

attack	Dragon	NFR	RealSecure	SessionWall	NetProwler
NTinfoscan	x	x	x	x	
pod		x	x		x
back	x	x	x	x	
httptunnel					
land	x	x	x	x	
secret					
ps					
portsweep		x	x	x	x
eject					
back			x		
loadmodule					
secret					
mailbomb		x			
ipsweep		x	x	x	x
phf	x		x	x	
httptunnel					
satan		x			
mailbomb		x			
perl					
ipsweep		x	x	x	x

Results: tcpreplay (cont.)

attack	Dragon	NFR	RealSecure	SessionWall	NetProwler
eject					
crashiis	x	x			
crashiis	x				
satan		x	x		
portsweep		x	x	x	x
neptune		x	x	x	x
secret					
perl					
land	x	x	x	x	
ipsweep			x	x	x
ftp-write	x	x			
phf	x	x		x	
perl					
ps					
pod		x	x	x	x
neptune		x			x
crashiis					
loadmodule					
perl					
ps					

Results: tcpreplay (cont.)

attack	Dragon	NFR	RealSecure	SessionWall	NetProwler
eject					
portsweep		x		x	x
ftp-write					
totals	9	20	15	14	10

Conclusions

- IDS will lack focus as a research topic until better measurement techniques are developed
  - quality metrics
  - testing processes
  - tools and their requirements
- Corollary: commercial IDS will not improve until freely-available tools exist to exploit their deficiencies
  - 1.5 years after the original SNI IDS paper was released, none of the vendors tested (except for NFR) had fixed their deficiencies, and no new IDS developed since its publication had addressed any of the issues either
  - 2 months after publishing fragrouter on BUGTRAQ, at least three vendors finally implemented IP fragment and TCP session reassembly
- We hope nidsbench is a step in the right direction

Future work

- Feeping creaturism
  - new fragrouter attacks (denial-of-service, etc.)
  - additional tcpreplay traffic shaping options and input file formats
- Integration into IDS testing frameworks

Exeunt

Nidsbench is freely available under a BSD-style license:

`http://www.anzen.com/research/nidsbench/`

Any questions?

Acknowledgments

The authors are grateful to Niels Provos, who originally suggested fragrouter's one-way router architecture, and to the primates at monkey.org for their help in testing our testing tools.

References

[SJ95] S. Dawson and F. Jahanian, "Probing and Fault Injection of Protocol Implementations," Proceedings of the International Conference on Distributed Computer Systems, May 1995.

[SJM97] S. Dawson, F. Jahanian, T. Mitton, "Experiments on Six Commercial TCP Implementations Using a Software Fault Injection Tool," University of Michigan Technical Report CSE-TR-298-96, 1997.

[RM98] R. Maxion, "Measuring Intrusion Detection Systems," Workshop on Recent Advances in Intrusion Detection, September 1998.

[hor98] horizon, "Defeating Sniffers and Intrusion Detection Systems," Phrack Magazine, Issue 54, Article 10, December 1998.

[KK99] K. Kendall, "A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems," Masters' thesis, MIT, June 1999.

[PN98] T. Ptacek and T. Newsham, "Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection,", Secure Networks, Inc., January 1998.

[GSAL98] H. Debar, M. Dacier, A. Wespi, and S. Lampart, "An Experimentation Workbench for Intrusion Detection Systems," Report RZ 2998, IBM Research, March 1998.

[Pax98] V. Paxson, "Bro: A System for Detecting Network Intruders in Real-Time," USENIX Security Symposium, 1998.

[PF97] V. Paxson, and S. Floyd, "Why We Don't Know How To Simulate The Internet," Proceedings of the Winter Simulation Conference, December 1997.

[UCSec96] N. Puketza, K. Zhang, M. Chung, B. Mukherjee, and R. Olsson , "A Methodology for Testing Intrusion Detection Systems,'' IEEE Transactions on Software Engineering, Vol. 22, No. 10, pp. 719-729, October 1996.