Testing Intrusion-Detection Systems: Claims & Evidence

Roy A. Maxion

Dependable Systems Laboratory
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
Email: maxion@cs.cmu.edu

IFIP 10.4 workshop on
Dependability and Survivability
29 June 2002

Hilton Head Island, South Carolina

Plan

- Brief overview of intrusion-detection systems
- Claims and the quality of evidence supporting them
- Discussion of selected aspects of testing/evidence:
  - Definitions
  - Data
  - Methodology
- Figures of merit for intrusion detection systems
- Focus on accuracy and impediments to it
- Examples from research & commercial literature
- Recommendations

Caveats

- This talk is about research systems reported in the technical literature, not about commercial systems.
- There is little public information about how commercial enterprises test their systems.
- The talk addresses observations about certain shortcomings in extant IDS testing efforts.
- Demonstration by example.

Three studies (for comparison)

- [Puketza] Puketza et al, 1996: A Methodology for Testing Intrusion Detection Systems
- [ATT] Schonlau et al, 2001: Computer Intrusion - Detecting Masquerades
- [Dragon] Mueller & Shipley, 2001: Dragon Claws its Way to the top.

What's being tested?

Intrusion-detection system (IDS):

A combination of hardware and software that monitors and collects system and network information and analyzes it to determine if an attack or an intrusion has occurred. [Allen et al., 2000]

Pertaining to techniques which attempt to detect intrusion into a computer or network by observation of actions, security logs, or audit data. Detection of break-ins or attempts either manually or via software expert systems that operate on logs or other information available on the network. [NSA glossary]

Two kinds of IDS

- Signature-based
  - Detection of previously seen "attacks"
  - Uses some form of pattern recognition
  - Reasonable hit rates, low false-alarm rates
  - Combines detection and diagnosis
- Anomaly-based
  - Detection of novel "attacks"
  - Uses some form of modeling of normal behavior
  - Typically high false-alarm rates
  - Detection of symptom (anomaly) only; diagnosis is a separate decision-making process (but often thought of (erroneously) as being part of the detection process)

How an IDS works

- Violation of a model of behavior.
- Models can be built by many methods:
  - Library of patterns (for signature-based systems)
  - Simple mean & standard deviation
  - Neural networks
  - Genetic algorithms
  - Markov chains
  - Uniqueness measures
  - Naïve Bayes
  - N-gram libraries

What is testing?

- To examine, observe or evaluate critically.
- To examine or judge concerning the worth, quality, significance, amount, degree, or condition of.
- To submit a claim to such conditions or operations as will lead to its proof or disproof or to its acceptance or rejection a test of a statistical hypothesis.
- ⇨ … conducted and described in such a way that other investigators can replicate (or at least judge) what has been done.

Test/evaluation goals

- Evaluations are decision-making aids.
- Therefore, they need to be:
  - Convincing, persuasive, dependable in themselves
  - Reliable - outcome is consistent over repeated observations; multiple observers will agree
  - Repeatable - outcome can be verified by independent observers
  - Valid - measures are true; metrics measure what they purport to measure; variables influence what they purport to influence
  - Objective - unbiased, lacking in systematic error
  - Defensible - claims are supported with evidence … as in a dependability/safety case

Dependability/safety case

- A documented (written) body of evidence that provides a convincing and valid argument that a system is adequately dependable/safe in a given environment. (Adapted from Bishop/Bloomfield 97 & 98.)

  A dependability case contains:

  - an explicit set of claims about the system;
  - a body of evidence to support those claims;
  - a set of arguments linking claims to evidence;
  - a clear list of the assumptions and judgements underlying the arguments.

Claims

- Make a specific claim about a property of the system or subsystem.
- This document is dependable; it can be used accurately as a ready reference for 90% of inquiries, for 95% of novice users, within 2 minutes of first contact.
- This intrusion-detection system will detect all attacks directed against it, with a detection latency of 2 seconds, and a diagnostic accuracy of 100%.

Types of claims

- Claims can be about any attribute of a system:
- Claims about security (from external attacks, or from masqueraders)
- Claims about … functional correctness, response time, maintainability, usability (by operators or clients), accuracy of results, robustness to overload, modifiability, upgradability, etc.
- Some claims (in intrusion detection) are implied.

Evidence

- Evidence should support the claim being made.
- Evidence can come from many sources:
  - the design (e.g., n-modular redundancy)
  - the development process (e.g., SEI level 5)
  - simulated experience (e.g., testing, benchmarking, etc.)
  - prior field experience (e.g., "a previous deployment worked well")

Forms of evidence

- Facts - based on scientific principles and prior research
- Assumptions - based on prior experience

What constitutes evidence?

- Formal proof
- Model checking
- Inspection
- Test

  ⇨ Extraordinary claims require extraordinary evidence.

Where is the evidence?

- Abstract
- Introduction
- Problem being solved
- Related work
- Approach
- ⇨ Method
- ⇨ Data
- Analysis
- Results
- Discussion
- Conclusion

Experimental method

- An experiment properly done should address (at least) these issues:
  - Definitions
  - Materials & apparatus
  - Subjects, data sets, workloads, algorithms, etc.
  - Metrics, measures & analyses
  - Experimental design (e.g., full factorial)
  - Procedure
  - Results
  - Discussion

IDS claims - figures of merit

- Speed
- Accuracy (in terms of detection & diagnosis)
- Performance / throughput / capacity
- Ease of use for operators & system administrators
- Detection/decision latency
- Attack (event) coverage
- Stress
- Resource usage
- Resiliency

Additional figures of merit

- Robust engines (do not crash)
- Timely & strong signature bases
- Remote manageability & management framework
- Scalability
- Usable interfaces
- Data mining & correlation functionality
- Cost effectiveness
- Customizability

From: [Dragon] Network Computing, August 2001

Accuracy

- Focus is on detection accuracy.
- Accuracy - freedom from [unquantified] error
- If an intrusion-detection system isn't accurate, then other aspects of its performance don't mean much.
- Accuracy includes not only hits, misses and false alarms, but also information about the kinds of errors made, as well as coverage boundaries.

Impediments to accuracy

There are many impediments, but we focus on these three, because they are so consistently found in IDS evaluations, and because they are so fundamental:

- Inadequate definitions
- Incompetent or "untruthful" data sets
- Faulty experimental methods

Accuracy and definitions

- Good definitions are essential in several aspects of an accurate evaluation, to wit:
  - Stating the goals of the evaluation.
  - Defining the system boundaries; e.g., is the entire system being evaluated, or only certain components?
  - Listing the system services (should be in the spec).
  - Delineating terms such as *slow* and *fast*, or *hard* and *easy*, or *expert* and *novice*.
- Other people will need to use these definitions in the same way you do, so operationalize all definitions.

Operational definitions

- Operational definitions tell users how data are collected. If the method or definition changes, the result changes. When users of data do not know how the data were collected, they easily make invalid assumptions, leading to incorrect interpretations, improper analyses, and erroneous conclusions.
- If definitions aren't made operational, you will essentially be saying, "I don't care how you do it - you make the decisions."
- Example: Measuring the height of school children.

Many fuzzy definitions in ID world

- Intrusion
  - 10 people, 10 definitions :-)
  - Privilege escalation? Break in, but do nothing?
- Intrusion detector (Tripwire?)
- Attack (port scan?)
- Exploit
- Vulnerability
- Real time
- Fault tolerance
- Stress
- Insider, outsider
- Accuracy ...

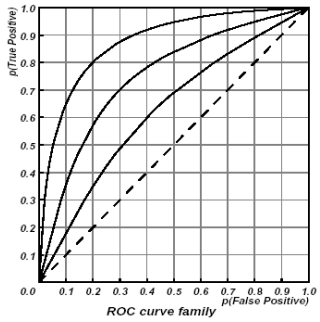
Defining accuracy measures

- Hits, misses, false alarms
- Drawn from signal-detection theory

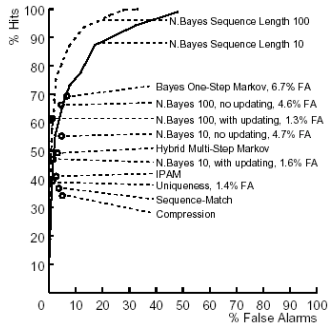
Signal detection theory

		Observation 0	Observation 1
Signal	0	Correct Rejections	False Alarms
	1	Misses	Hits

Receiver operating characteristic (ROC)



Example of empirical ROC curves



Bogus ROC curves

- Example of poor or missing definitions.
- ROC from one well-known experiment (LL-98) showed false alarms per day on the X-axis.
- This violates the definition of ROC.
- FA/day is not useful because the data rate was not provided.
- Two consequences:
  - Misleading conclusions could be drawn
  - Other people unwittingly make same kind of mistake

How they fared: definitions

- [Puketza]
  - Terms defined, but not operationalized
- [ATT]
  - Terms well defined
  - Good enough to know that there were problems
- [Dragon]
  - Terms not defined

Reminder ...

- Definitions are important.
- Even if a definition is wrong, it's critical to state what one thinks it is; others can correct it.

Accuracy and data

- Accuracy can't be measured unless reference is made to some basis of comparison, or oracle.
- This oracle is often called ground truth … a statement of undisputed fact.
- Ground-truth data are needed for:
  - estimating environmental & workload factors (Maxion/Tan 00)
  - constructing models of normal behavior (Schonlau 01)
  - mapping detector performance regions (Tan/Maxion 02)
  - scoring evaluation competitions (LL-98)

Ground-truth data

- VERY hard to acquire, but worth the trouble
- Subject to the same definitional requirements as terminology
- Example from ATT/Schonlau 01

Accuracy and methodology

- The method by which an experiment is conducted can change the outcome.
- Fault-injection methodology, in particular, needs to be done carefully so that the objectives of the experiment are realized, e.g.,
  - Which of several classifiers/detectors is better?
  - What kinds of errors do detectors make, and why?

Illustrative example

- Masquerader study … "irregularities"
  - Data-gathering methodology
  - Experimental procedure

Masquerader definition

- Colloquially, masquerading is…
  the act of substituting oneself for another.
- To masquerade is…
  - To disguise
  - To assume the appearance of something one is not
  - To furnish with a false appearance or an assumed identity
  - To obscure the existence or true state or character of something
- These variants characterize the computer masquerader, a user who impersonates another user, usually with mischievous intent.
- A special case is the **insider**, a legitimate user who substitutes one *mission* for another.

The masquerader question

- There are two forms of the masquerader question.

Given an unidentified sample of user behavior:

- Is the unknown sample a match for a particular user, or not?
- Which user, of a given population, does the sample match?
- Our work focuses on the first question.

Sample of user commands

sed	netscape	fecc
eqn	slogin	driver
troff	ssh-add	purify.s
dpost	xauth	sh
echo	sh	make
sh	ls	array_te
sh	ssh	ex
sh	scp	expr
cat	head	hostname
netstat	gzip	id
netscape	tar	nawk
troff	ls	getopt
tbl	configur	ppost
sed	fec	awk
sed	be	ppost
eqn	ld64_	bc
troff	driver	date
dpost	sendmail	cat
echo	mailx	sed
sh	ksh	cat
gs	sendmail	cat
Ghostvie	Touch	postprin

Description of the data

- Output of Unix acct auditing package
- Keyboard commands (truncated - without command-line arguments) extracted from Unix process accounting logs (not directly from the shell)
- 70 users; 15,000 commands each
- 50 users selected for test community
- Collected over a period of several days to several months (details not given)
- Users generated their 15,000 commands at different rates; days to months (details not given)
- Comprised of user names and command sequences
- Arguments to commands are omitted due to privacy concerns (but flags and shell grammar (e.g., pipes) also omitted)
- Commands grouped into 150 blocks of 100 (chosen for convenience)
- First 50 blocks (5,000 commands) are uncontaminated
- Starting at block 51, masquerade blocks may randomly contaminate
- Some commands not typed explicitly by users: Shell, make, .profile
- Size of alphabet: 635 in training data / 856 in testing data
- Max symbols used by any one user: 138
- Min symbols used by any one user: 5

Methodology from prior work

- Fault (masquerader) injection:
  - .1 probability of injection, followed by .8 probability of same masquerader; else reset to .1
- Advantages:
  - Attempts to replicate natural circumstances (not too many masqueraders)
- Disadvantages:
  - Results are not fair to detection algorithms, because particular user/masquerader combinations may, by chance, be especially fortunate or unfortunate; coincidental interactions may over-represent detection success or failure.
- Normal user data used as proxy for masquerader data.
- Not all users were injected with masqueraders.
- Each user was not subjected to same masqueraders.
- Doesn't facilitate error analysis.

Bad things in the data

- Multiple masquerade events were largely drawn from the same masquerader.
- No user was injected with data from more than three different masqueraders.
- 15-20% of hits were achieved by repeatedly identifying blocks taken from the same masquerader.
- Choosing the masquerader-user pairs differently might have had a significant impact on the success profiles reported.
- Using data drawn randomly from normal users as masquerader data has its perils.
- The masquerade events embedded in the data of User 10 consist of 13 adjacent blocks of a single command, i.e., one long sequence of 1300 "popper" commands.
- The relatively strong performance of every algorithm against User 10, except compression, is therefore hardly surprising.

Two experimental methods

- ATT configuration
  - As described
- 1v49 configuration
  - General method: Train on UserX 5000; test on 5000 * 49 (nonself) plus 10,000 minus injections of users 51-70 (self).
  - No profile of nonself, because the training data of the other 49 users is used as testing data.

Methods ranked by cost: *M + FA*

Method	Hits	Misses	FA	Cost
Bayes 1-Step Markov	69.3	30.7	6.7	37.4
Naïve Bayes (no updating)	66.2	33.8	4.6	38.4
Naïve Bayes (updating)	61.5	38.5	1.3	39.8
Hybrid Markov	49.3	50.7	3.2	53.9
IPAM	41.1	58.9	2.7	61.6
Uniqueness	39.4	60.6	1.4	62.0
Sequence Matching	36.8	63.2	3.7	66.9
Compression	34.2	65.8	5.0	70.8

Ranking: *Cost = Misses + False Alarms*

Methods ranked by cost: *M + 6*FA*

Method	Hits	Misses	FA	Cost
Naïve Bayes (updating)	61.5	38.5	1.3	46.3
Naïve Bayes (no updating)	66.2	33.8	4.6	61.4
Uniqueness	39.4	60.6	1.4	69.0
Hybrid Markov	49.3	50.7	3.2	69.9
Bayes 1-Step Markov	69.3	30.7	6.7	70.9
IPAM	41.1	58.9	2.7	75.1
Sequence Matching	36.8	63.2	3.7	85.4
Compression	34.2	65.8	5.0	95.8

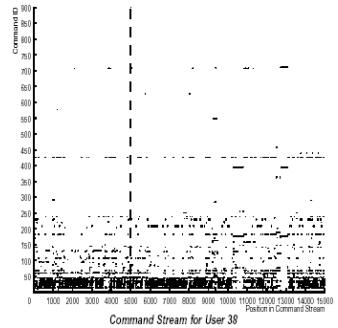
Ranking: *Cost = Misses + 6 * (False Alarms)*

- 1% false alarms puts uniqueness at the top of the ATT ranking, implying that uniqueness is their best method. For this to be the case, a false alarm must cost 6 times as much as a miss.
- The range of coefficients for the false alarm term for which the whole ranking will be the same as that observed with a coefficient of 6, is 5.71 to 7.05.

What makes a successful masquerader?

- Based on confusion matrix ... number of masquerade blocks that were undetected.
  - Who are the top masqueraders?
    - 38: 1649/2450 = 67%
    - 05: 1614/2450 = 66%
    - 25: 1575/2450 = 64%
  - How successful are they?
    - Successful more than 59% of the time: more than half of their 50x49 = 2450 masquerade attempts were successful.
    - Top masqueraders are successful across the board: more than half the users falsely accepted 35 or more of their blocks as self.
  - What makes them successful?
    - High use of popular shared tuples (User 38)
    - Moderate use of very popular shared tuples (Users 37, 31)
    - Not using any command too often
    - Not using less popular tuples highly frequently

User 38: good masquerader



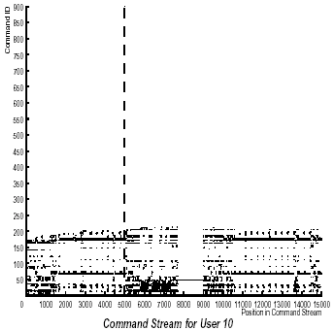
Command Stream for User 38

- Best masquerader
- Most shared tuples

What causes false alarms?

- Concept drift (nonstationarity).
- User's behavior changes after the profile is built (and surpasses updating).
- Block boundaries (100 commands) inconsistent / incompatible with behavior changes.

User 10: most false alarms



Command Stream for User 10

- Updating doesn't help here.
- Squiggles fall across block boundaries; don't update as a block.
- Extreme discontinuities between training and testing data (extreme nonstationarity).

Shared-tuple hypothesis

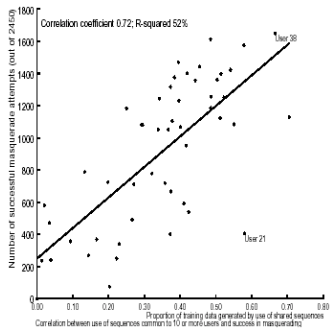
- There is a correlation between:
  - Number of shared tuples in a user's data;
  - User's success as a masquerader.

The more shared tuples, the greater the user's success as a masquerader.

Shared-tuple hypothesis (cont.)

- Shared tuples
  - Definition: Sequence of at least two commands which occurred in the data of at least 10 users; overlaps were removed.
  - The longest such sequence contained 49 commands, and was shared by 18 out of 50 users.
  - Shared sequences account for 34.75% of the training data, across all 50 users.
  - The amount of training data accounted for by shared tuples ranges between 0 and 70.5%.
  - Shared tuples may arise from common scripts built into the computing environment. The following 10-tuple occurred multiple times in the data of 27 users: hostname, id, nawk, getopt, true, true, grep, date, lp, find.

Correlation: Masquerader success & shared tuples



- 38: Best masquerader
- 21: Poor masquerader
  - Large portion of shared tuples (37% one 5-gram)

Removing Sharing Tuples

- 72 distinct sequences were shared by at least 10 users.
  - Together, these sequences accounted for 34.75% of the old training data.
- The longest sequence contained 49 commands and was shared by 18 users.
- The following 10-tuple occurred multiple times in the data of 27 users:
  - hostname, id, nawk, getopt, true, true, grep, date, lp, find
- Removing these tuples entirely led to a drastic reduction in the amount of training data available for certain users, so experiment was based on just 1000 commands of training data.

Action with respect to Shared Tuples			
	Left intact	Condensed	Removed
Hits %	68.8	75.0	76.2
Misses %	32.2	25.0	23.8
False Alarms %	NA	NA	NA

Table 1: Detupled Data Experiments

Enriched command-line data

Truncated	Enriched
cd	cd cpsc504
more	more susan.lst
diff	diff susan.lst julie.lst
lpr	lpr –Pjp susan.lst
setenv	setenv TERM amb amb
rwho	rwho -a
set prompt	set prompt = "VAXC{!} [$cwd:t] --> "
nroff	nroff –me proposal | more
ls	ls –F –l ~candym/.em* | -l ~candym/.em*
enscript	enscript -2Gr -L66 -Palw -h *.c print66 *.c

What's different about the new data set?

- 168 users versus 70
- Users grouped according to experience

Novice	Experienced	Computer Scientist	Non-programmer
13	15	21	1

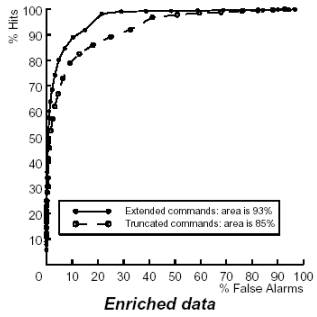
- Data captured from shell, not log - no scripts
- Additional information
  - All commands in command-line, not just first one
  - Arguments to the commands
  - Flags
  - Grammar (pipes, semi-colons, etc.)
  - Whether command was alias, and if so, for what
  - Directory from which command was issued
  - ID of xterm in which command was issued
  - Date
  - Use of history
  - Record of errors made
- Information about who took how long to produce how much data

Results Naïve Bayes, no updating, block size 10

Data	Old	Old	New	New
Type of data	**Truncated**	**Truncated**	**Truncated**	**Enriched**
Amount of training data	5,000	1,000	1,000	1,000
Hits %	47.1	69.2	67.4	82.1
Misses %	52.9	30.8	32.6	17.9
False Alarms %	1.6	12.1	4.7	5.7
Cost (equal weights)	54.5	42.9	37.3	23.6
Cost (FA =6*Miss)	62.5	103.4	60.8	52.1

Table 2: Average results for NB on old and new data

ROC curve: extended vs. truncated data



Lessons learned

- Classifier
  - Naïve Bayes performs well, but it's not clear whether its failings are due to intrinsic shortcomings in the classifier or in the data.
- Concept drift
  - Is a serious influence on false alarms.
- Shared tuples
  - Shared tuples (scripts) are a serious problem for *insider* masquerader detection.
- Data truncation
  - The poor performance of every method tried so far on this data indicates that truncated command-line data alone is not enough to profile a user.
  - Useful additions might be: arguments to commands; type and length of sessions; user categories (job-type, department), etc.

How they fared: data & methodology

- Data
  - [Puketza]
    - High-level descriptions only; no details
  - [ATT]
    - Adequate details, but incomplete
    - Left open assumptions about data-gathering methodology
  - [Dragon]
    - No details; even high-level descriptions were lacking
  - [Other]
    - Misleading definition of average AFB
    - Cannot predict effects of environment
- Methodology
  - [Puketza]
    - No details; cannot replicate or judge
  - [ATT]
    - Adequate details, but not suited to purpose
    - Good enough to recognize unsuitability and to effect remedy
  - [Dragon]
    - No details; cannot replicate or judge

Is IDS testing different/harder? (no)

- Too many degrees of freedom, due to ingenuity of human hackers
- Continuously changing environment
- No single agreed-upon datastream that IDSs use
- IDSs must detect many things as opposed to just one thing
- Different IDSs are focused on different types of attacks
- The IDS community is trying to get a one-number metric
- Hard to characterize the attack space - > 3700 vulnerabilities
- Hard to collect attack scripts
- Hard to get victim software
- No taxonomy of faults and their manifestations
- Testing requirements are different between signature-based and anomaly-based IDS systems
- Testing requirements different for network/host-based systems

Recommendations

- Test in ideal environment.
  - Determines what detector can/can't see.
- Explore limits; perform sensitivity analyses
  - What is scope of exploit/technique? Can family of allowable manifestations be characterized?
- Test under stress
  - Influence of background traffic & its content
- Define & characterize carefully & thoroughly
- Attend to data gathering/manipulation methods
- Support claims with evidence