My IDS is better than yours!

or... is it ?

Stefano Zanero

Ph.D. Candidate, Politecnico di Milano

CTO & Founder, Secure Network S.r.l.

Outline

- Establishing a need for testing methodologies
  - Testing for researchers
  - Testing for customers
- Requirements for IDS testing
  - Some theory here... sorry !
- State of the art
  - Academic test methodologies
  - Industry test methodologies (?)
- Recommendations and proposals

The need for testing

- Two basic types of questions
  - Does it work ?
    - If you didn't test it, it **doesn't work (but it may be pretending to)**
    - **Includes: security testing**
      - **Do I really need to rant about antiviruses?**
  - **How well does it work ?**
    - **Objective criteria**
    - **Subjective criteria**

Researchers vs. Customers

- What is testing for researchers ?
  - – Answers to the “how well” question in an objective way
  - – Scientific = repeatable (Galileo, ~1650AD)
- What is testing for customers ?
  - – Answers to the “how well” question in a subjective way
  - – Generally, very custom and not repeatable, esp. if done on your own network

Relative vs. absolute

- Absolute, objective, standardized evaluation
  - Repeatable
  - Based on rational, open, disclosed, unbiased standards
  - Scientifically sound
- Relative evaluation
  - “What is better among these two ?”
  - Not necessarily repeatable, but should be open and unbiased as much as possible
  - Good for buy decisions

IDS requirements and metrics

- A good test needs a definition of **requirements and metrics**
  - Requirements: “does it work ?”
  - Metrics: “how well ?”
  - I know software engineers could kill me for this simplification, but who cares about them anyway ? :)
- Requirements and metrics are not very well defined in literature & on the market, but we will try to draw up some in the following
- Let me focus for simplicity on Network IDS

False positives and negatives?

- Should alert on intrusions
  - False Negative issue
  - Polymorphism and evasion
  - “zero-day” detection (duh !)
  - Traffic overload and packet loss
- Should not alert on non intrusions
  - False positives issue

Testing IDS (naïve approach)

- Simulated attack data
  - Execution of exploits (as downloaded from the usual sources)
- Real-world sample traffic
  - Intermixed with execution of exploits
- Examples of the concept:
  - LL/MIT evaluations (with truth files!)
  - Other data sets (UCSB Treasure Hunt, Defcon CTF) lack a truth file
- **Anyway it's simple: count false positives and false negatives ! Isn't it ?!**

NO !!!
It's awfully
complicated !

Anomaly vs. Misuse

- Describes normal behaviour, and flags deviations
- Can recognize any attack (also 0-days)
- Depends on the model, the metrics and the thresholds
- Statistical alerts

- Uses a knowledge base to recognize the attacks
- Can recognize only attacks for which a "signature" exists
- Depends on the quality of the rules
- Precise alerts

Misuse Detection Caveats

- It's all in the rules
  - Are we benchmarking the *engine or the ruleset ?*
    - *Badly written rule causes positives, are they FP ?*
    - *Missing rule does not fire, is this a FN ?*
      - *How do we measure coverage ?*
    - *Correct rule matches attack traffic out-of-context (e.g. IIS rule on a LAMP machine), is this a FP ?*
      - *This form of tuning can change everything !*
  - *For commercial IDSs this may make little sense, but for Snort makes a lot of sense*
- *A misuse detector alone will* ***never catch a zero-day attack, with a few exceptions***

Anomaly Detection Caveats

- No rules, but this means...
  - *Training*
    - *How long do we train the IDS ? How realistic is the training traffic ?*
  - *Testing*
    - *How similar to the training traffic is the test traffic ? How are the attacks embedded in ?*
  - *Tuning of threshold (more on this later)*
- *Anomaly detectors:*
  - *If you send a sufficiently strange, non attack packet, it will be flagged. Is that a “false positive” for an anomaly detector ?*

An issue of polimorphism

- Computer attacks are polimorph
  - *So what ? Viruses are polimorph too !*
    - *Viruses are as polimorph as a **program** can be, attacks are as polimorph as a **human** can be*
  - *Good signatures capture the vulnerability, bad signatures the exploit*
- *Plus there's a wide range of:*
  - *evasion techniques*
    - *[Ptacek and Newsham 1998] or [Handley and Paxson 2001]*
  - *mutations*
    - *see ADMmutate by K-2, UTF encoding, etc.*

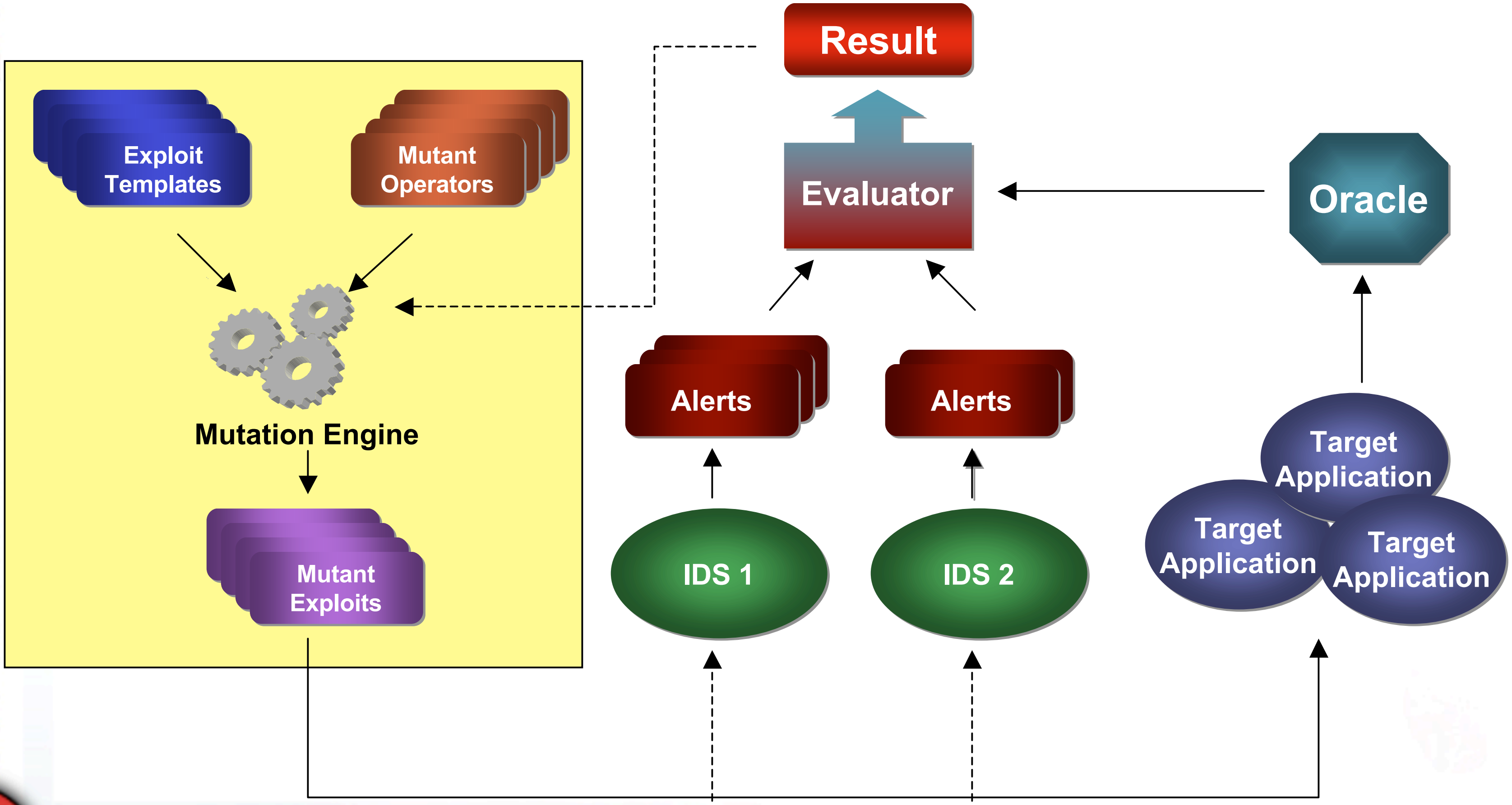
Evaluating polimorphism resistance

- Open source KB and engines
  - Good signatures should catch key steps in exploiting a vulnerability
    - Not key steps of a particular exploit
  - Engine should canonicalize where needed
- Proprietary engine and/or KB
  - Signature reverse engineering (signature shaping)
  - **Mutant exploit generation**

Signature Testing Using Mutant Exploits

- **Sploit implements this form of testing**
  - **Developed at UCSB (G.Vigna, W.Robertson) and Politecnico (D. Balzarotti - kudos)**
    - Generates mutants of an exploit by applying a number of mutant operators
    - Executes the mutant exploits against target
    - Uses an oracle to verify the effectiveness
    - Analyzes IDS results
- Similar earlier efforts
    - **Thor (R. Marty), much more limited**
    - **AGENT (Rubin, Jha, and Miller): formal, based on logical induction for mutations**

Architecture of Sploit



Examples of mutations

- Network and transport layer
  - Ipv6, IP Fragmentation, TCP/UDP Splitting
- Session and presentation layer
  - SSL, RPC
- Application-layer
  - Protocol rounds, protocol-specific techniques (e.g. for FTP, HTTP...)
- “Exploit-layer”
  - Shellcode mutations, alternate encodings

Example of Results

	Snort			ISS RealSecure		
Exploit	Baseline Attack	Mutated Attack	Evasion Technique	Baseline Attack	Mutated Attack	Evasion Technique
WUFTP	Detected	Evaded	Telnet ctrl seq Shellcode IP splitting	Detected	Evaded	Telnet ctrl seq Shellcode
WUIMAP	Detected	Evaded	Zero prefix Shellcode	Detected	Evaded	Junk char insertion
IISDD	Detected	Detected		Detected	Evaded	HTTP evasion
DCOMRPC	Detected	Detected		Detected	Detected	
IISUNI	Detected	Evaded	URL encoding	Detected	Evaded	HTTP evasion
ISSNSLOG	Detected	Detected		Detected	Evaded	HTTP evasion
ISSISAPI	Detected	Detected		Detected	Evaded	HTTP evasion
WSFTP	Detected	Evaded	Telnet ctrl seq IP splitting	Detected	Evaded	Telnet ctrl seq
SSLMSKEY	Detected	Evaded	SSL Null record	Detected	Evaded	SSL Null record
HTTPCNK	Detected	Evaded	HTTP evasion	Detected	Evaded	HTTP evasion

Comments on Sploit

- A great idea, with some intrinsic limits:
  - Tests engine and signatures together
  - Qualitative, more than quantitative
  - Strongly dependent on exploit and mutation templates quality and selection
    - We could bias the test result, if used to compare two different intrusion detection systems
- In conclusion, Sploit is great for testing IDSs and their rulebase, but has some limits if used as an evaluation or comparison tool

Measuring Coverage

- *If ICSA Labs measure coverage of anti virus programs ("100% detection rate") why can't we measure coverage of IDS ?*
  - *Well, in fact ICSA is thinking about it... see https://www.icsalabs.com/icsa/main.php?pid=jgh475fg*
  - *Problem:*
    - *we have rather good zoo virus lists*
    - *we do not have good vulnerability lists,let alone a reliable wild exploit list*
- *We cannot **absolutely** measure coverage, but we can perform **relative** coverage analysis (but beware of biases)*

How to Measure Coverage

- **Offline coverage testing**
  - **Pick signature list, count it, and normalize it on a standard list**
  - **Signatures are not always disclosed**
  - **Cannot cross compare anomaly and misuse based IDS**
- Online coverage testing
  - We do not have all the issues but
  - How we generate the attack traffic could somehow influence the test accuracy

False positives and negatives

- Let's get back to our first idea of "false positives and false negatives"
  - All the issues with the definition of false positives and negatives stand
- Naïve approach:
  - Generate realistic background traffic
  - Superimpose a set of attacks
  - Feed the test data to the IDS
  - Compare IDS alerts with attacks, mark false positives & false negatives
- We are all set, aren't we ?

Background traffic

- Too easy to say “background traffic”
  - Use real data ?
    - Realism 100% but not repeatable or standard
    - Privacy issues
    - Good for relative, not for absolute, eval
  - Use sanitized data ?
    - Sanitization may introduce statistical biases
      - e.g. character distribution in sanitized packets
    - Network peculiarities may induce higher DR
    - The more we preserve, the more we risk
  - In either case:
    - Attacks or anomalous packets could be present!

Background traffic (cont)

- So, let's really generate it
  - Use “noise generation” ?
    - Algorithms depend heavily on content, concurrent session impact, etc.
  - Use artificially generated data ?
    - Approach taken by DARPA, USAF...
    - Create testbed network and use traffic generators to “simulate” user interaction
    - This is a good way to create a repeatable, scientific test on solid ground
  - Use no background.... yeah, right
  - What about broken packets ?
    - http://lcamtuf.coredump.cx/mobp/

Attack generation

- Collecting scripts and running them is not enough
  - How many do you use ?
  - How do you choose them ?
  - Do you use evasion ?
  - You need to run them against vulnerable and not vulnerable machines
  - They need to blend in perfectly with the background traffic
- Again: most of these issues are easier to solve on a testbed

Datasets or testbed tools ?

- Diffusion of datasets has well-known shortcomings
  - Datasets for high speed networks are huge
  - Replaying datasets, mixing them, superimposing attacks creates artefacts that are easy to detect
    - E.g. TTLs and TOS in IDEVAL
  - Tcpreplay timestamps may not be accurate enough
    - Good TCP anomaly engines will detect it's not a true stateful communication
- Easier to describe a testbed (once again)

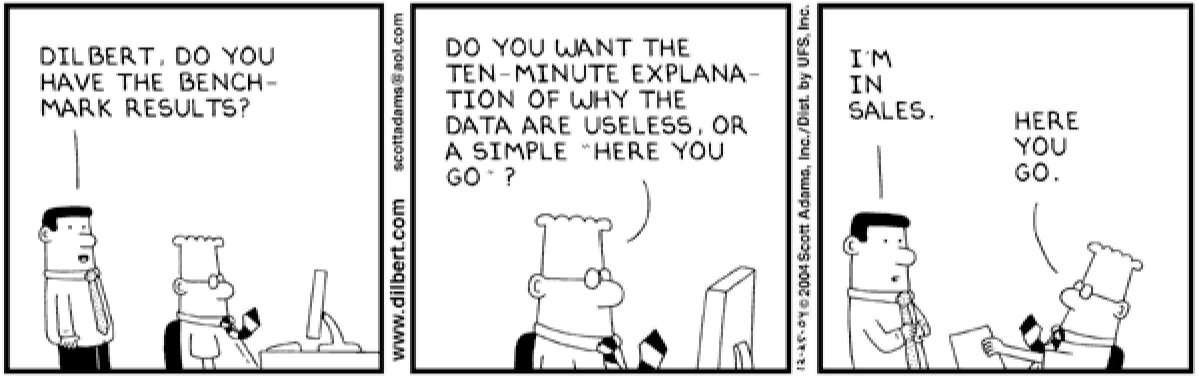
Generating a testbed

- We need a realistic network...
  - Scriptable clients
    - We are producing a suite of suitable, GPL'ed traffic generators (just ask if you want the alpha)
      - Scriptable and allowing for modular expansion
      - Statistically sound generation of intervals
      - Distributed load on multiple slave clients
  - Scriptable or real servers
    - real ones are needed for running the attacks
    - For the rest, Honeyd can create stubs
  - If everything is FOSS, you can just describe the setup and it will be repeatable !
    - **Kudos to Puketza et al, 1996**

Suitable traffic mixes

- Q: “What does the Internet look like ?”
  A: “How am I supposed to know ?!”
  - Measurements: CAIDA, Cisco, our own
  - All agree & disagree but some trends are:
    - TCP is predominating (up to 95% on bytes, 85 to 90 on packets); UDP 5-10% P, ICMP 1-2% P
    - HTTP dominant (75% B, 70% P for CAIDA, over 60% P for Cisco, 65% P in our environment) but in slight decreasing trend
    - DNS, SMTP (5-8% P even) account for most of the rest of Internet traffic; NNTP and FTP declining; on “general” networks gaming and peer-to-peer traffic can reach 10%
    - Average packet size ~570 byte, many full-size

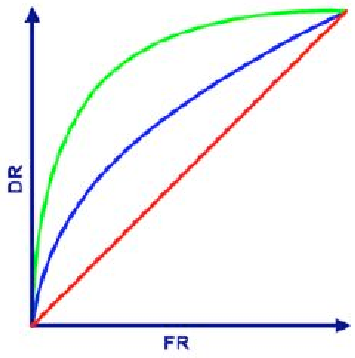
Do raw numbers really matter?

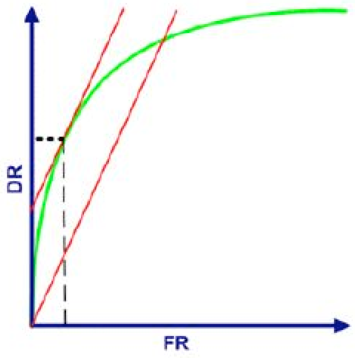

DILBERT, DO YOU HAVE THE BENCHMARK RESULTS?
DO YOU WANT THE TEN-MINUTE EXPLANATION OF WHY THE DATA ARE USELESS, OR A SIMPLE "HERE YOU GO"?
I'M IN SALES.
HERE YOU GO.
www.dilbert.com
scottadams@aol.com
© 2004 Scott Adams, Inc./Dist. by UFS, Inc.

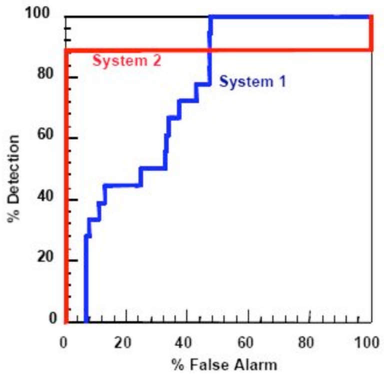
© UFS, Inc.

- If Dilbert is not a source reliable enough for you, cfr. Hennessy and Patterson
    - Personally, I prefer to trust Dilbert... kudos to Scott Adams :-)
- Raw numbers seldom matter in performance, and even less in IDS

ROC curves, then !







- Great concept from signal detection, but:
  - they are painful to trace in real world
  - they are more meaningful for anomaly IDS than misuse IDS
    - Depends, again, on definition of false positive

It is written “performance”...

- But it reads like “speed”
  - If you want to measure “how fast” an IDS is, you once again need to define your question
    - Packets per second or bytes per second (impacts NIC capacity, CPU, and memory bus speed)
    - Number of hosts, protocols and concurrent connections (memory size and memory bus speed, CPU speed)
    - New connections per second (memory bus speed, CPU speed)
    - Alarms per second (memory size, CPU speed, mass storage, network, whatever...)
  - Each metric “measures” different things !

Load-testing IDS (2)

- Using TCP replay devices then ?
  - At high speeds, buffer size issues requires to use more replay interfaces
  - Sync issues as well as aggregation issues
  - As we said... traces are not really good for many reasons on stateful devices

Network issues

- Network worries also:
  - Traffic generators, attack network, victim network are connected to a switch
  - Span port capacity could limit the IDS
    - On a Gb Ethernet port inter-packet arrival gap is 96 ns...
    - If multiple Fast Ethernet ports, generating ~80Mbps, are used, multiple frames will happen in a 96 ns bucket
    - Port buffer fills up = the **switch** drops packets
    - In real conditions, a choke point (such as a router) will reduce the likelyhood for this to happen

Metrics, metrics

- Once again: what to measure?
  - Throughput ? Delay ? Discarded packets ?
  - Connections/sec or packets/sec ?
  - In theory, this thing acts like an M/M/1/c finite capacity queue...
    - Arrival process is Poisson (simplification, it actually isn't)
    - Service time is exponential (another simplification, could be load-dependent)
    - There is a finite buffer c (this is realistic)
  - Rejection rate can be statistically computed

Is this really M/M/1/c ?

- We have a stateless traffic replication tool named “Blabla” which we use for simple tests
  - On commodity hardware can generate up to 50kps average (100Mb) following an exp distribution with great accuracy
- We tested Snort 2.x and an old release of Cisco IDS
  - Raw numbers are meaningless, but they always are
- Both behave as M/M/1/c systems
  - Mostly, adding open connections make them load-dependent
  - Like any M/M/1/c system, the c parameter is a trade-off between discarded packets and congestion (i.e. waiting time)
    - This is important for moving a system in-line !
    - It's not easy to tune on the Snort/Libpcap/Linux stack
    - In our experiments, c turns out to be “small”

Queues quirks

- The queueing model also says...
  - That traffic distribution matters !
  - That packets/connections/open connections ratios matter !
  - Packets/bytes ratio matters !
  - We have also verified, as others showed before, that types of packets, rules and checks impact on the service times
- So, all these things should be **carefully documented in tests... and you should read them when evaluating other people tests**

Existing tests I'm aware of

- A bit outdated
  - Puzetzka at UC Davis (oldies but goldies)
  - IBM Zurich labs (God knows)
  - IDEVAL (more on this later)
  - AFRL evaluations (cool, but not open)
- Current tests (2002-2003...)
  - NSS group tests
    http://www.nss.co.uk
  - Neohapsis OSEC
    http://osec.neohapsis.com/
  - Miercom Labs/Network World
    http://www.networkworld.com/reviews/2002/1104rev.html

MIT/LL and IDEVAL

- IDEVAL is the dataset created at MIT/LL
  - Only available resource with synthetic traffic and full dumps + system audit files
  - Outdated systems and attacks
  - Very few attack types, in particular host-based IDS have just basic overflows...
  - Well known weaknesses in NIDS data:
    - TTLs, TOS, source IP, ... all detectable
  - IDEVAL has been used by **each and every researcher in the field (including me), i.e. it has biased all the research efforts since 1998**

NSS Tests

- NSS Group tests are perhaps the most famous industry testing ground
- On the whole, not bad, but:
  - They are non repeatable (since attacks and other parameters are unspecified)
    - Being not really scientific and not really based on a specific scenario, what's their aim
  - Include lots of qualitative evaluations
  - Use either noise or HTTP traffic for stress testing
  - Unspecified distribution characters of traffic
  - Aging attacks and evasions (for what we know)

Neohapsis / OSEC

- A new pretender on the block
- Good idea, an open, repeatalbe methodology, but:
  - Not addressing breadth of KB
  - Use either noise or HTTP traffic for stress testing
  - Unspecified distribution characters of traffic
  - Not really suitable for anomaly based products

Miercom/Network World

- Less known than the others
- More journalistic than scientific
- Yet, a very good description of the setup, the attacks, and the testing conditions
  - Still not addressing breadth of KB
  - Still HTTP traffic for stress testing
  - Still unspecified distribution characters of traffic
  - But a very very good testing methodology indeed

Conclusions

- Testing IDS is a real, huge mess
  - But still, we must do something
- We are still far away from designing a complete, scientific testing methodology
  - But we can say a lot of things on wrong methodologies
- You can try to design customer-need driven tests in house
  - Difficult, but the only thing you can do
- In general, beware of those who claim “My IDS is better than yours”

QUESTIONS ?

Thanks for your attention !!!

Feedback/Followup/Insults welcome
stefano.zanero@polimi.it
s.zanero@securenetwork.it

Feel free to browse research, papers and presentations
http://www.elet.polimi.it/upload/zanero

Bibliography

- **Traffic measurements, internet traffic mixes**
  - K. Claffy, G. Miller, K. Thompson: The Nature of the Beast: Recent Traffic Measurements from an Internet Backbone http://www.caida.org/outreach/-papers/1998/Inet98/ (1998)
  - S. McCreary, K. Claffy: Trends in Wide Area IP Traffic Patterns: A View from Ames Internet Exchange. http://www.caida.org/outreach/papers/2000/-AIX0005/ (2000)
- **Polimorphism resistance testing**
  - **G. Vigna, W. Robertson, D. Balzarotti: Testing Network-based Intrusion Detection Signatures Using Mutant Exploits, ACM CCS 2004**
  - **S. Rubin, S. Jha, B. P. Miller: Automatic Generation and Analysis of NIDS Attacks, ACSAC 2004.**
- **General performance literature**
  - **D. Patterson, J. Hennessy: Computer Organization and Design: the Hardware/Software interface, 3rd ed., Morgan-Kauffman**

Bibliography (2)

- General IDS testing literature
  - M. Hall, K. Wiley: Capacity Verification for High Speed Network Intrusion Detection Systems http://www.cisco.com/en/US/products/hw/vpndevc/ps4077/prod_technical_reference09186a0080124525.html
  - M. J. Ranum: Experiences benchmarking Intrusion Detection Systems, http://www.snort.org/docs/Benchmarking-IDS-NFR.pdf
  - N. Athanasiades, R. Abler, J. Levine, H. Owen, G. Riley: Intrusion Detection Testing and Benchmarking Methodologies, 1st IEEE International Information Assurance Workshop, 2003
  - P. Mell,V. Hu, R. Lippmann, J. Haines, M. Zissman: An Overview of Issues in Testing Intrusion Detection Systems, NIST – LL/MIT, 2003
  - N. J. Puketza, K. Zhang, M. Chung, B. Mukherjee, R. A. Olsson: A Methodology for Testing Intrusion Detection Systems, IEEE Transactions on Software Engineering, 1996

Bibliography (3)

- IDEVAL and MIT efforts
  - K. Kendall: A database of computer attacks for the evaluation of intrusion detection systems. Master's thesis, MIT, 1999
  - R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, K. Das: Analysis and results of the 1999 DARPA off-line intrusion detection evaluation. Proceedings of the 3rd International Workshop on Recent Advances in Intrusion Detection, Springer-Verlag
  - John McHugh: Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln laboratory. ACM Trans. on Information and System Security, 3(4):262-294, 2000.
  - M. V. Mahoney, P. K. Chan: An analysis of the 1999 DARPA / Lincoln laboratory evaluation data for network anomaly detection. In Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection (RAID 2003), Pittsburgh, PA, USA, September 2003.