UNIVERSITY OF TRENTO

Network Security

AA 2020/2021

System hardening

(Intrusion detection systems, honeypot)

Function of an IDS

- Firewalls prevent unwanted access to network resources that should be isolated w.r.t. another network
- IDS monitors connections/activity on a host
- Intrusion Prevention Systems (IPS) can act over “malicious” behaviour
- IDS → passive monitoring
- IPS → active monitoring
- In reality functionalities are not entirely distinct
  - Commercial lingo rather than actually different technology

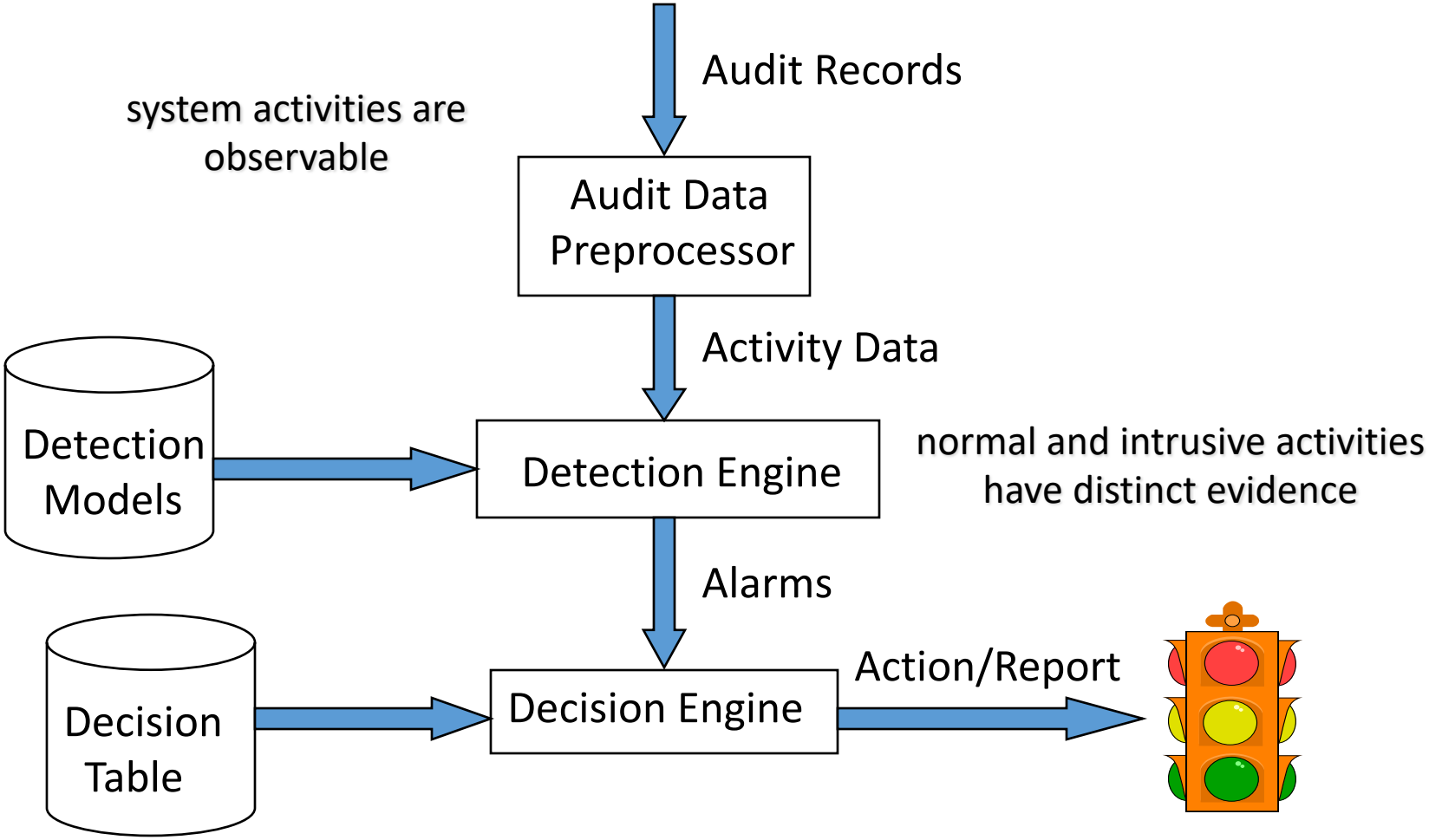
Definitions

- Intrusion
  - A set of actions, occurring on your system, aimed to compromise the security goals, namely
    - Integrity, confidentiality, or availability, of a computing and networking resource
- Intrusion detection
  - The process of identifying and responding to intrusion activities

Intrusion Detection Systems

- classify intrusion detection systems (IDSs) as:
  - Host-based IDS: monitor single host activity
  - Network-based IDS: monitor network traffic
- logical components:
  - sensors - collect data
  - analyzers - determine if intrusion has occurred
  - user interface - manage / direct / view IDS

Components of Intrusion Detection System



IDS Requirements

- run continually
- be fault tolerant
- resist subversion
- impose a reasonable overhead on system
- configured according to system security policies
- adapt to changes in systems and users
- scale to monitor large numbers of systems
- provide graceful degradation of service
- allow dynamic reconfiguration

Efficiency of IDS system

- Accuracy: the proper detection of attacks and the absence of false alarms
- Performance: the rate at which traffic and audit events are processed
  - To keep up with traffic, may not be able to put IDS at network entry point
  - Instead, place multiple IDSs downstream
- Fault tolerance: resistance to attacks
  - Should be run on a single hardened host that supports only intrusion detection services
- Timeliness: time elapsed between intrusion and detection

Intrusion Detection Approaches

- Modeling
  - Features: evidences extracted from audit data
  - Analysis approach: piecing the evidences together
    - Misuse detection (a.k.a. signature-based)
    - Anomaly detection (a.k.a. behavioural-based)
    - Specification-based detection

Misuse detection

- IDS equivalent of “default allow” policies
- ”blacklist” patterns that are believed to be related to malicious activities
  - System calls
  - Payloads in network protocols
- Signature-based
  - Very diffused detection technique
  - Easy to deploy
  - Typical implementation for network-based IDSs
- As all blacklisting approaches (signature-based) it can only detect patterns that are *already known*

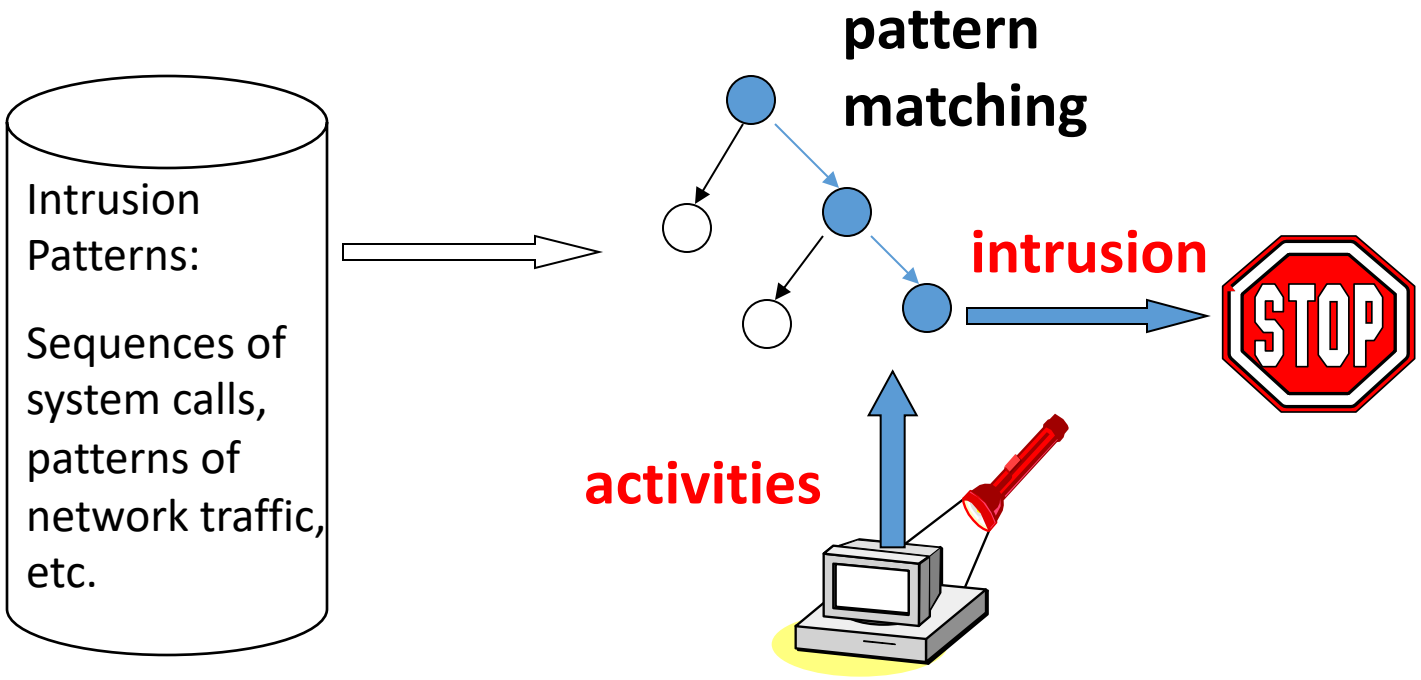
UNIVERSITY OF TRENTO

Signature Detection

- observe events on system and applying a set of rules to decide if intruder
- approaches:
  - rule-based anomaly detection
    - analyze historical audit records for expected behavior, then match with current behavior
  - rule-based penetration identification
    - rules identify known penetrations / weaknesses
    - often by analyzing attack scripts from Internet
    - supplemented with rules from security experts

Step	Command	Comment
1.	`%cp /bin/csh /usr/spool/mail/root`	– assumes no root mail file
2.	`%chmod 4755 /usr/spool/mail/root`	– make setuid file
3.	`%touch x`	– create empty file
4.	`%mail root < x`	– mail root empty file
5.	`%/usr/spool/mail/root`	– execute setuid-to-root shell
6.	`root%`	

Misuse Detection



Example: *if* (traffic contains "x90+de[^\r\n]{30}") *then* "attack detected"
Advantage: Mostly accurate. But problems?

Can't detect new attacks

Anomaly detection

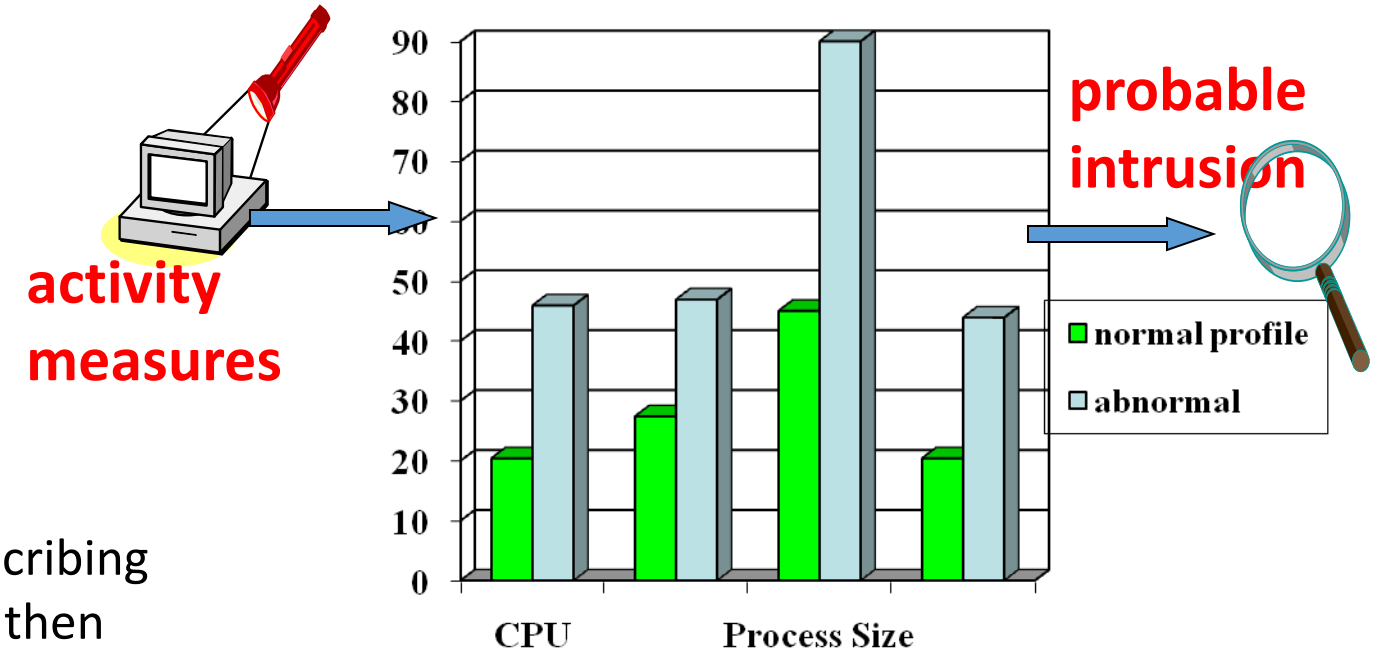
- Assumes intruder behaviour differs from legitimate profile
- Building legitimate profile may be an issue
  - Depends on data used for profiling (e.g. sampled vs whole dataset)
  - Profile can evolve → new “legitimate activity” looks suspicious
- Can be used both for HIDS and NIDS
  - HIDS → syscall, system file hashing, system states, ..
  - NIDS → protocol analysis, similar to application proxy
    - Monitoring as opposed to filtering]

Anomaly Detection

- threshold detection
  - checks excessive event occurrences over time
  - alone a crude and ineffective intruder detector
  - must determine both thresholds and time intervals
- profile based
  - characterize past behavior of users / groups
  - then detect significant deviations
  - based on analysis of audit records
    - gather metrics: counter, guage, interval timer, resource utilization
    - analyze: mean and standard deviation, multivariate, markov process, time series, operational model
    - model as a classification problem, machine learning classifiers

UNIVERSITY OF TRENTO

Anomaly Detection



Define a profile describing "normal" behavior, then detects deviations. Thus can detect potential new attacks.
Any problem ?

Relatively high false positive rates

- Anomalies can just be new normal activities.
- Anomalies caused by other element faults
  - E.g., router failure or misconfiguration, P2P misconfig
- Which method will detect DDoS SYN flooding ?

Specification-based detection

- Manually develop specifications that capture legitimate (not only previous seen) system behavior.
- Any deviation from it is an attack
- Pros
  - can avoid false-positive since the specification
  - can capture all legitimate behavior.
- Cons
  - hard to develop a complete and detailed specification, and error-prone.
- Approach:
  - state machine, extended finite state automata (EFSA)
  - augment FSA with state variables

example

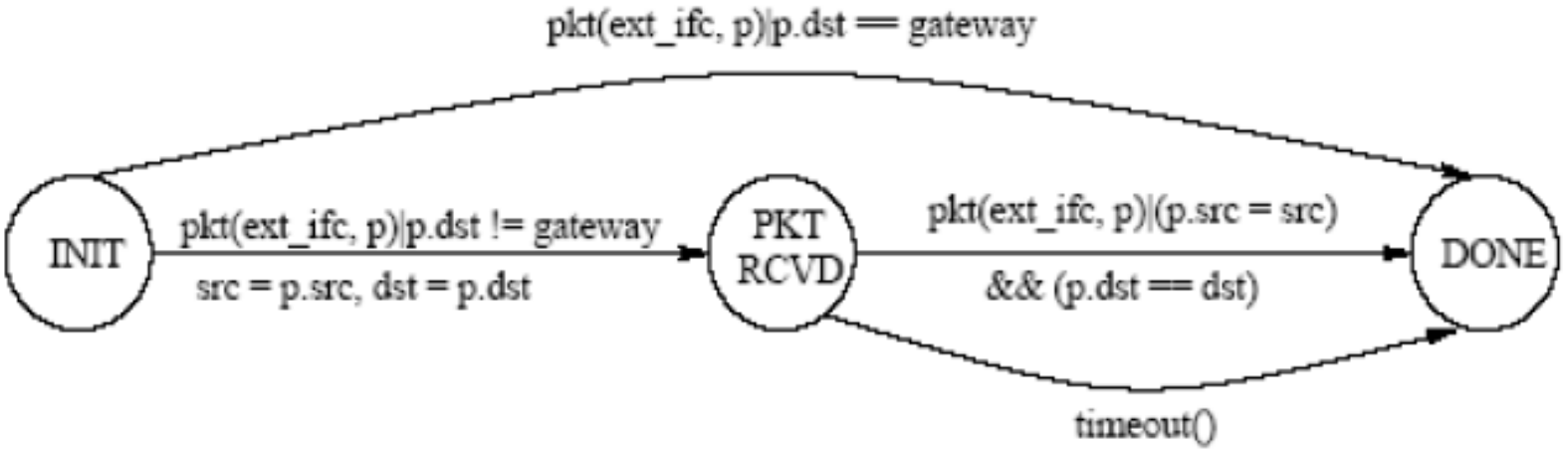
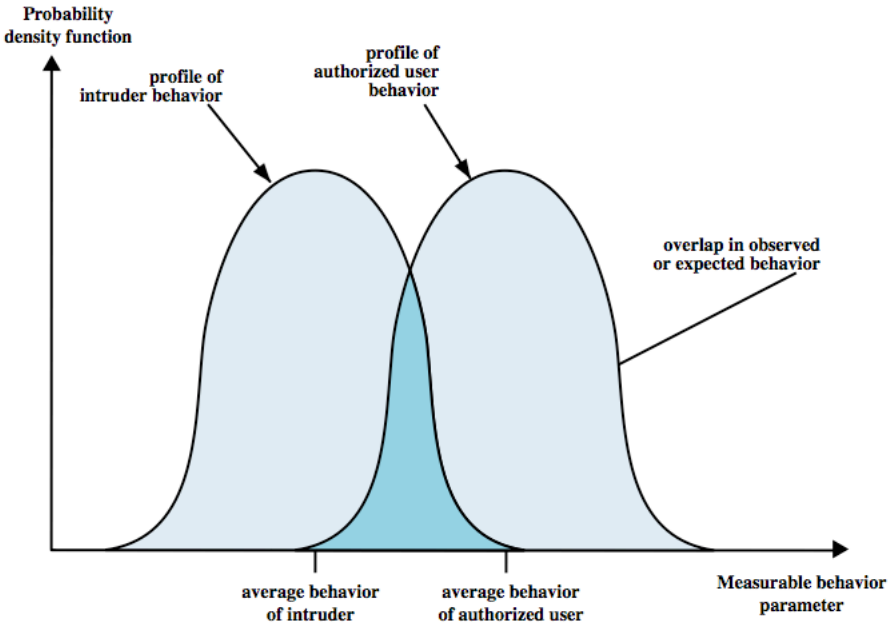


Figure 1: Simplified IP Protocol State Machine

IDS Principles

- assume intruder behavior differs from legitimate users
  - expect overlap as shown
  - observe deviations from past history
  - problems of:
    - false positives
    - false negatives
    - must compromise



The base-rate fallacy – or, can we have actually good detection rates?

- Both anomaly and misuses detection necessarily lead to false positives and false negatives
- A NIDS with 99% true positive rate and 99% true negative rate seems to have high-reliability alarms
  - → an alarm fires up → you should worry
  - → no alarm fires up → all is good
  - But is it?
- Base-rate fallacy
  - Simple derivation from Bayes theorem
  - Very well known by medics and doctors
  - Still making its way through in InfoSec

The base-rate fallacy [Axelsson 2000]

- Tests with high true positives and negatives rates yield much “worse” results than expected by the average user
- Remember Bayes theorem

$$P(A|B) = \frac{P(A) \cdot P(B|A)}{\sum_{i=1}^{n} P(A_i) \cdot P(B|A_i)}$$

This is P(B) expanded to all “n” cases for A that B comprises

- Let’s make the classic medical example
  - Attack = illness
  - IDS Alarm = medical test

Base-rate fallacy example

$$P(A|B) = \frac{P(A) \cdot P(B|A)}{\sum_{i=1}^{n} P(A_i) \cdot P(B|A_i)}$$

- A=event is *patient is sick*
- B=medical test says patient is sick
- P(A|B) = patient is actually sick given that test said so
  - Equivalent to "there is an actual attack given that NIDS fired alarm"
- Set TP=99%; TN=99% → P(B|A) = 0.99
- Diseases are rare. Say 1/10.000 people have the illness → P(A)=1/10.000
  - Most network traffic is legitimate

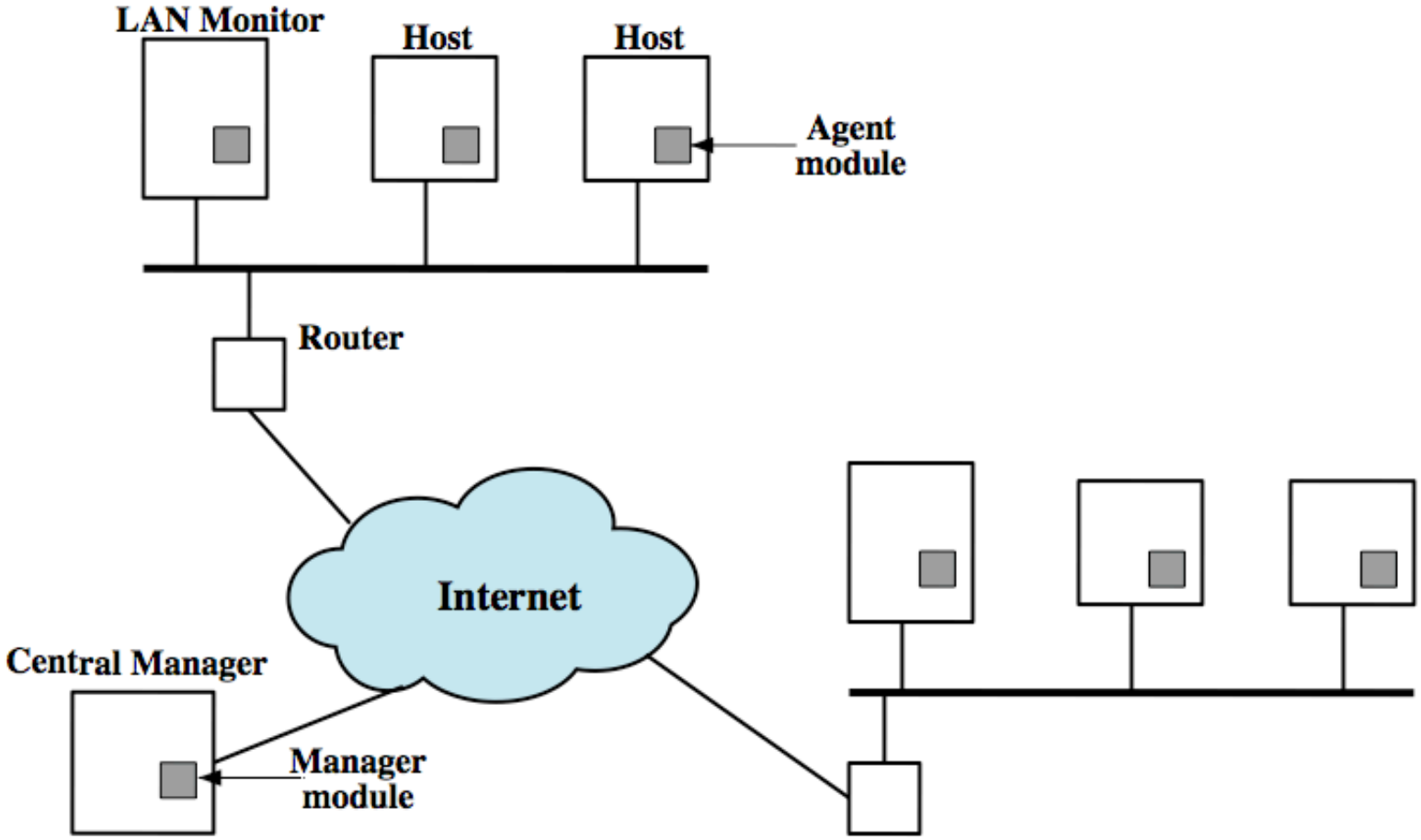
$$P(A|B) = \frac{1/10000 \cdot 0.99}{1/10000 \cdot 0.99 + (1 - 1/10000) \cdot 0.01} = 0.00980... \approx 1\%$$

- There is only 1% chance that patient is sick when test says so
  - An alarm is not very meaningful → IDS alarms are hard to manage → log analysis

Host-Based IDSs

- Use OS auditing and monitoring/analysis mechanisms to find malware
  - Can execute full static and dynamic analysis of a program
    - Monitor shell commands and system calls executed by user applications and system programs
  - Has the most comprehensive program info for detection, thus accurate
- Problems:
  - Only local view of the attack
  - If attacker takes over machine, can tamper with IDS binaries and modify audit logs

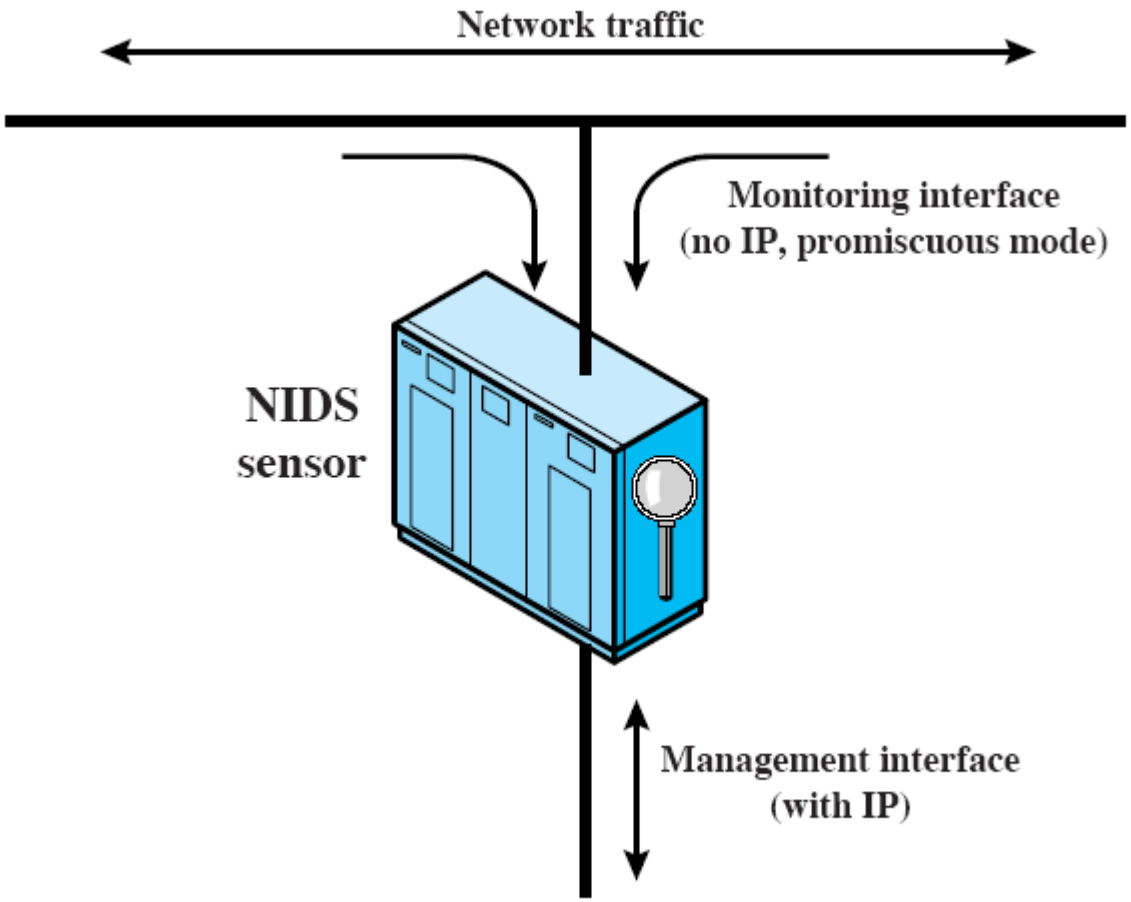
Distributed Host-Based IDS



Network-Based IDS

- network-based IDS (NIDS)
  - monitor traffic at selected points on a network
  - in (near) real time to detect intrusion patterns
  - may examine network, transport and/or application level protocol activity directed toward systems
- comprises a number of sensors
  - inline (possibly as part of other net device)
  - passive (monitors copy of traffic)

Passive NIDS Sensor



Network IDS

- Deploying sensors at strategic locations
  - For example, Packet sniffing via *tcpdump* at routers
- Inspecting network traffic
  - Watch for violations of protocols and unusual connection patterns
  - Look into the packet payload for malicious code
- Limitations
  - Cannot execute the payload or do any code analysis !
  - Even DPI gives limited application-level semantic information
  - Record and process huge amount of traffic
  - May be easily defeated by encryption, but can be mitigated with encryption only at the gateway/prox

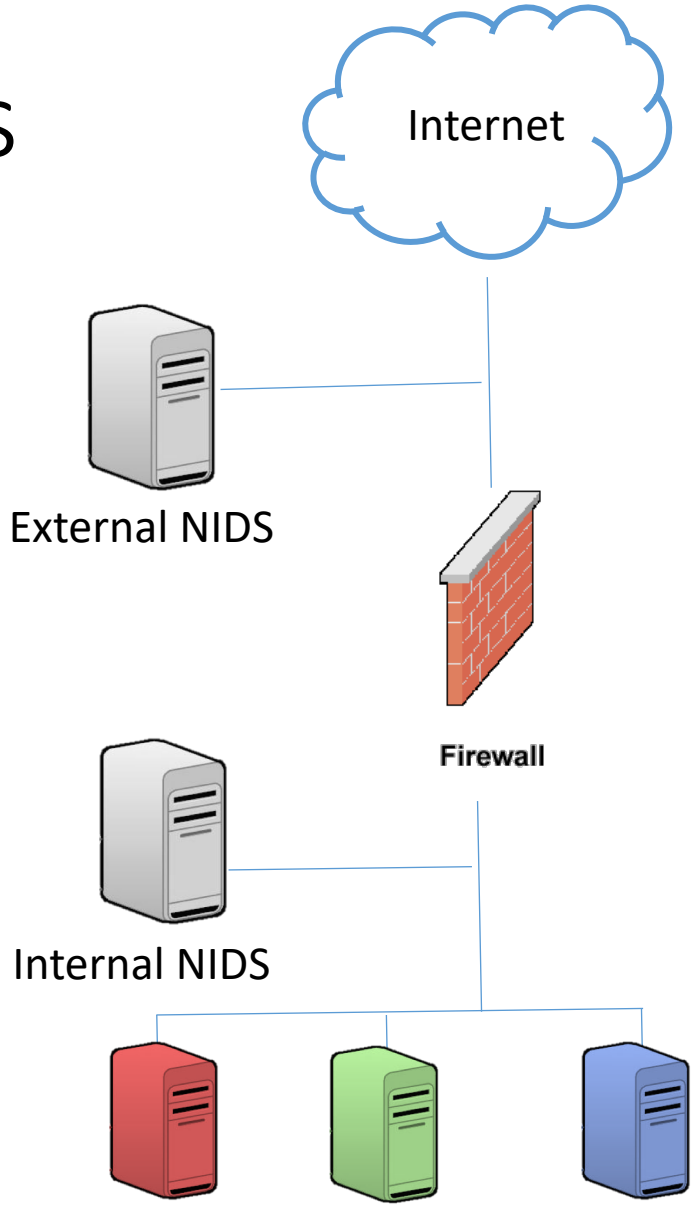
Host-based vs. Network-based IDS

- Give an attack that can only be detected by host-based IDS but not network-based IDS

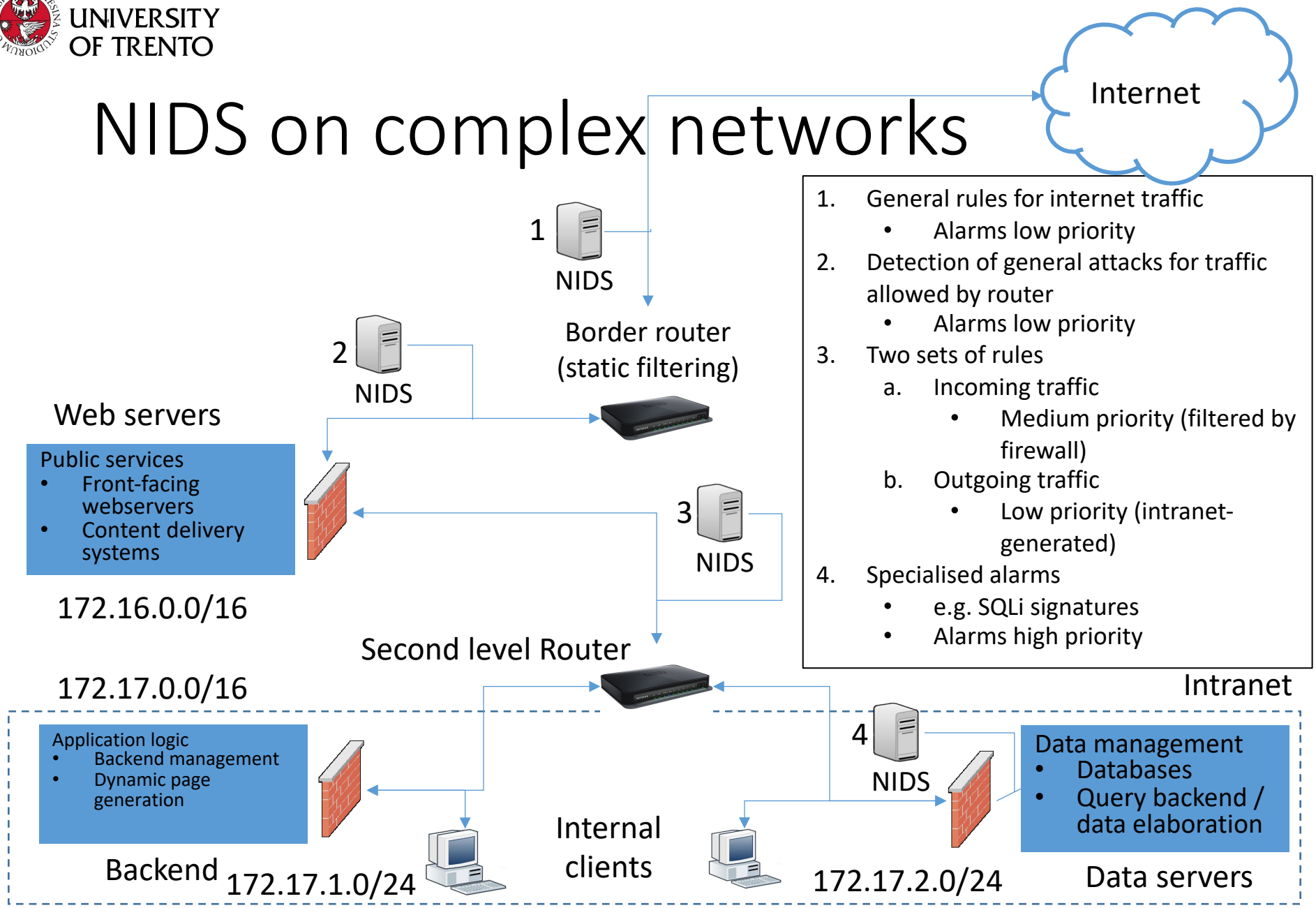
- Can you give an example only be detected by network-based IDS but not host-based IDS ?

Architectural aspects

- External NIDS
  - Analysis of all set of incoming traffic
  - Only general signatures are possible
    - high incidence of FP
  - All detected “attempted attacks” are logged
  - "normal" Internet traffic may generate many alarms
- Internal NIDS
  - Analysis of traffic allowed by the firewall
  - More specific signatures are possible
    - e.g. based on services behind firewall, subnet characteristics, ..
  - Says nothing about attacks attempted but blocked by firewall



NIDS on complex networks



1. General rules for internet traffic
   - Alarms low priority
2. Detection of general attacks for traffic allowed by router
   - Alarms low priority
3. Two sets of rules
   a. Incoming traffic
      - Medium priority (filtered by firewall)
   b. Outgoing traffic
      - Low priority (intranet-generated)
4. Specialised alarms
   - e.g. SQLi signatures
   - Alarms high priority

Example: SNORT signatures

- https://www.snort.org/

- Rule is a single line, rule header: everything before parenthesis, rule option: what's in the parenthesis.

- Syntax for rule header:

```
rule_action protocol src_add_range src_prt_range
dir_operator dest_add_range dest_prt_range
```

Ex. alert tcp 192.168.1/24 1:1024 ➔ 124.17.8.1 80

rule actions; alert, drop, log….

protocol; tcp, udp, icmp,…

direction: -> and <>

src,dst port ranges.

Example: SNORT signatures

```
alert icmp $EXTERNAL_NET any ➔ $HOME_NET any
(msg: “ICMP PING NMAP; dsize: 0; itype:8;)
```

- Rule generates alert for ICMP having empty payload, ICMP type 8, and arriving from the outside.
- This is part of an NMAP ping

```
alert tcp $EXTERNAL_NET any ➔ $HOME_NET 139
(msg: “DOS SBMdie attack”:; flags: A+;
content:”|57724c6568004577a|”;)
```

- Rule generates alert if a TCP packet from outside contains |57724c6568004577a| in payload and is headed to port 139 (netbios) for some internal host.
- This is part of a buffer overflow attack on a computer running Server Message Block Service.

Example: SNORT signatures

```
alert tcp $HOME_NET 2589 -> $EXTERNAL_NET any
(msg:"MALWARE-BACKDOOR - Dagger_1.4.0";
flow:to_client,established; content:"2|00 00 00 06
00 00 00|Drives|24 00|"; depth:16;
metadata:ruleset community; classtype:misc-
activity; sid:105; rev:14;)
```

dsize: payload size

offset: XX; (start at byte XX in payload)

depth: YY; (stop at byte YY in payload)

Network IDS

- Baseline implementation is of type *misuse detection*
  - Easier to implement
  - Network traffic is hard to predict even on well-controlled environments
- Signature example:

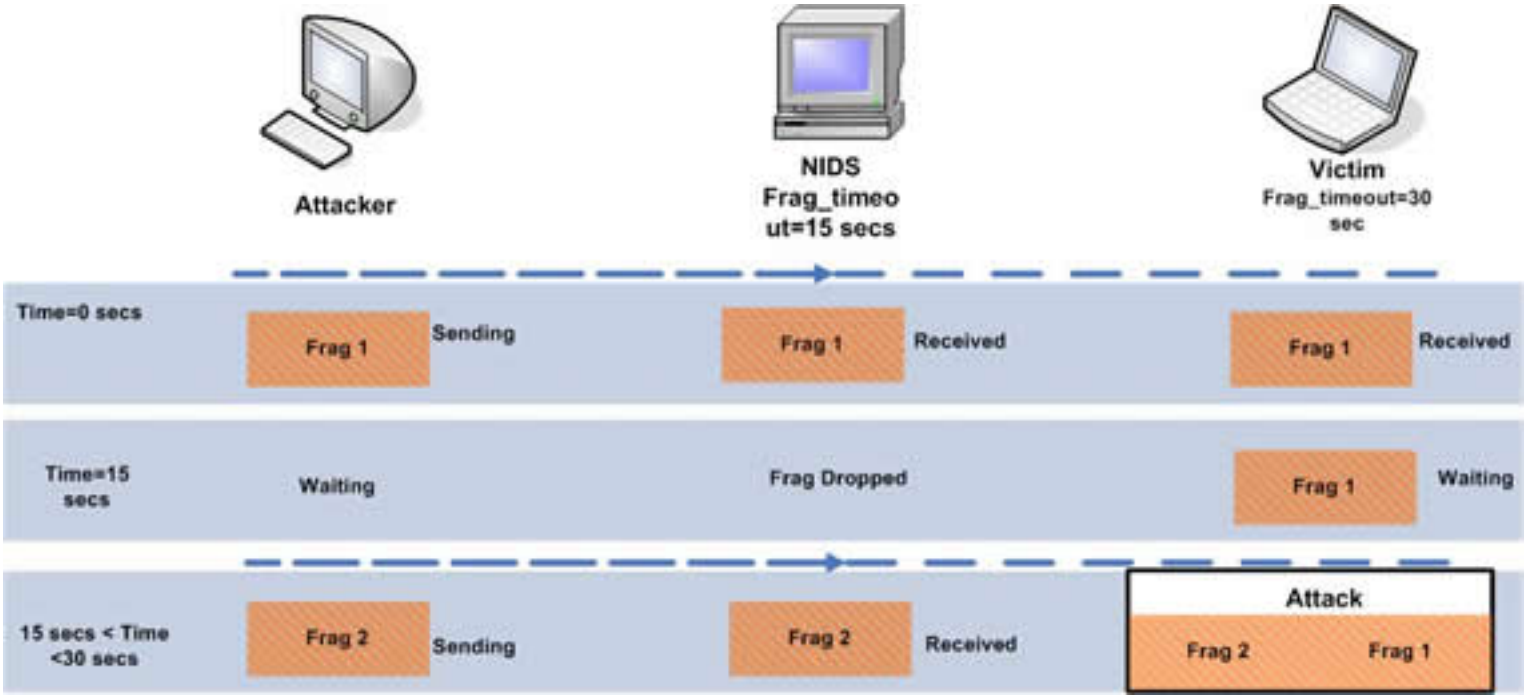
```
alert
tcp $EXTERNAL_NET any -> $HOME_NET 139
flow:to_server,established
content:"|eb2f 5feb 4a5e 89fb 893e 89f2|"
msg:"EXPLOIT x86 linux samba overflow"
reference:bugtraq,1816 reference:cve,CVE-1999-0811
```

NIDS evasion [Siddharth 2005]

- Signature-based evasion can be fairly trivial
- Depends on implementation of actual signature
  `content:"/bin/bash"`
  - → detects remote calls to bash
  - Does not detect string "/etc/../bin/bash", etc.
- More advanced techniques are typically based on IP fragmentation
  - All techniques have common goal: *NIDS sees different packet than client*
  - Look at these keeping in mind you may want to prevent the attacker from performing
    - Network mapping
    - OS fingerprinting

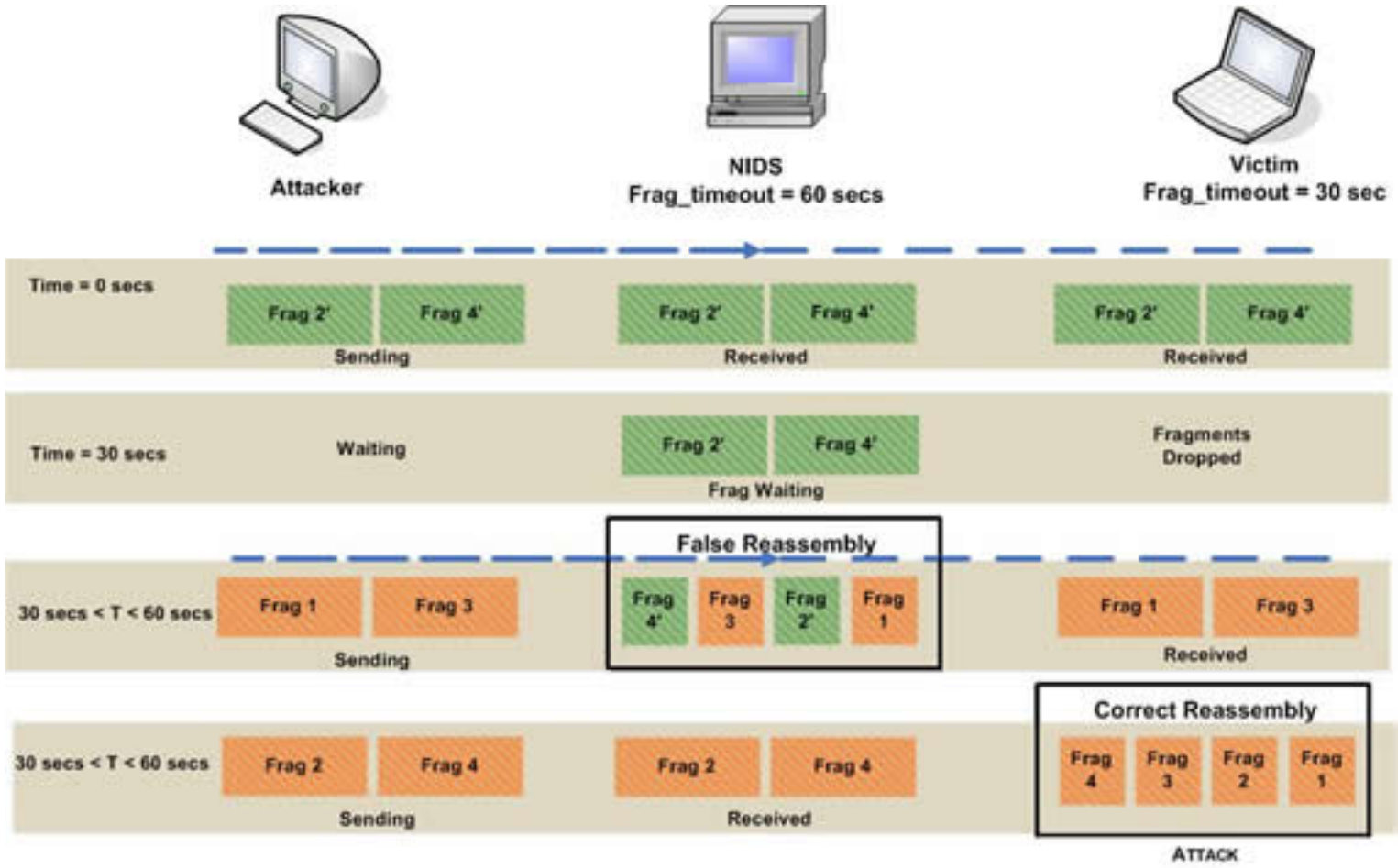
Evasion technique – Reassembly time-out

- NIDS has lower reassembly timeout than receiving client



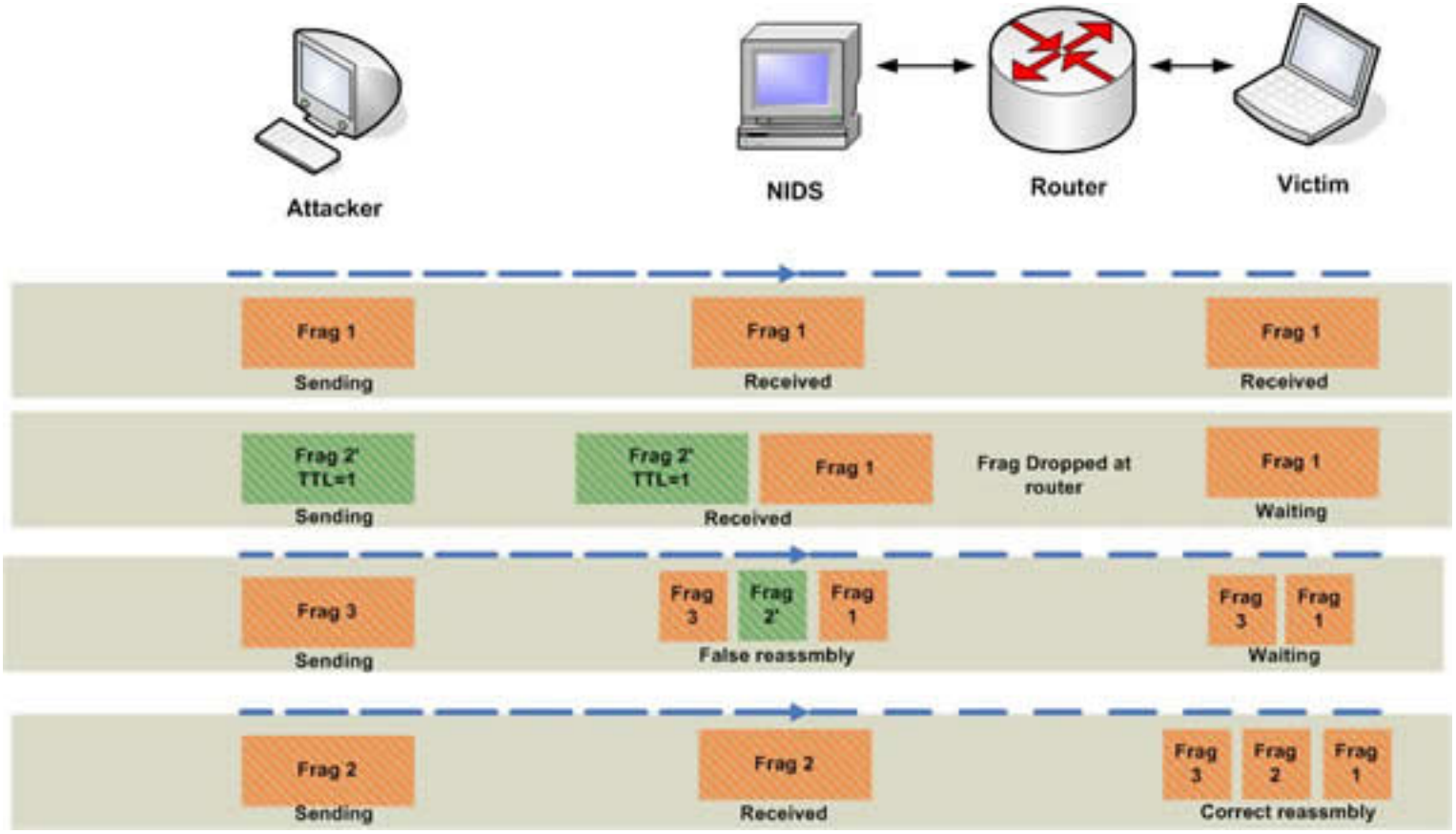
Evasion technique – Reassembly time-out (2)

- NIDS has <u>higher</u> reassembly timeout than receiving client



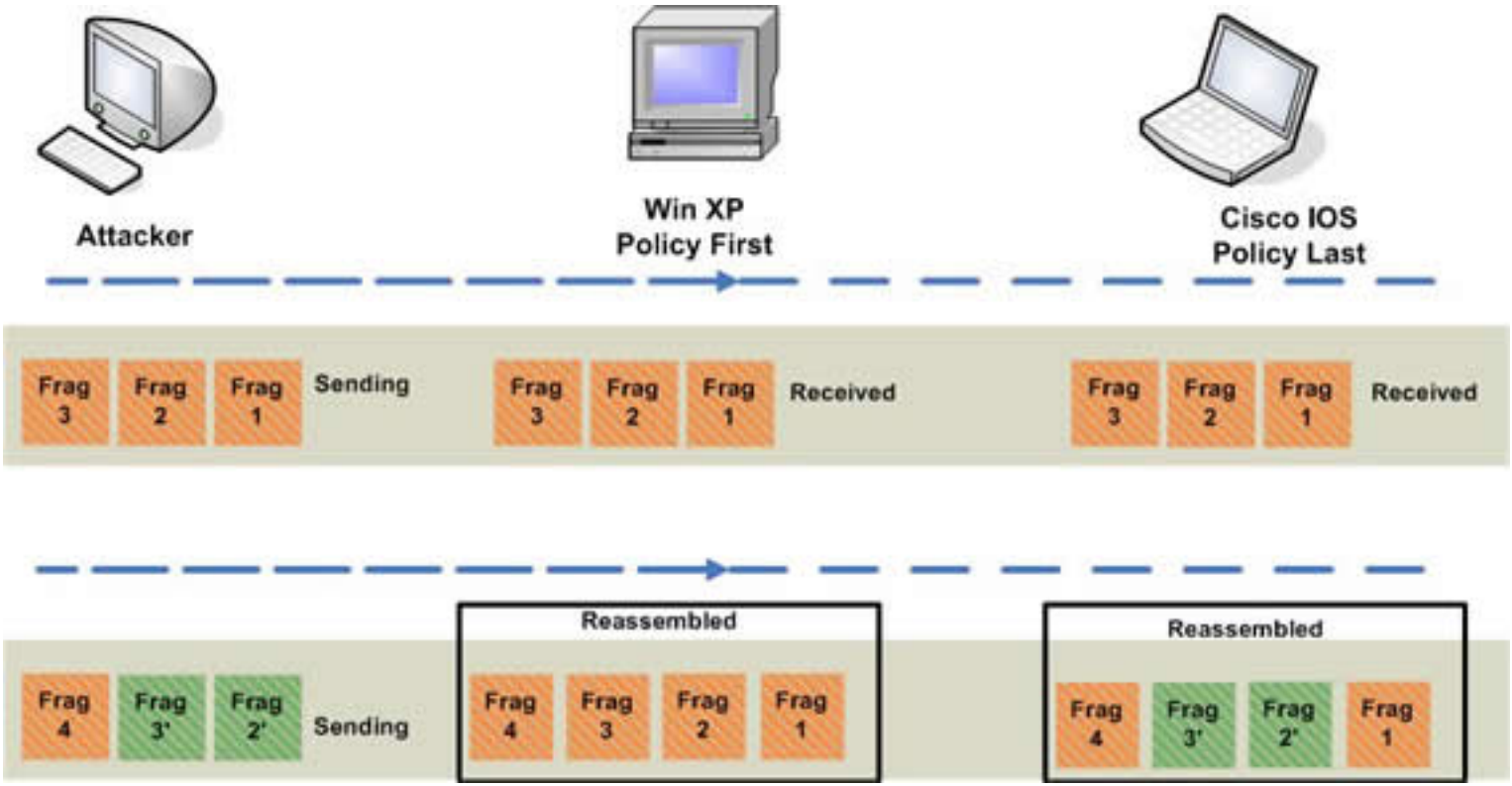
Evasion technique – Time-to-live

- Router drops packet analysed by NIDS that will not be delivered to victim



Evasion technique – Fragment replacement

- Some operating systems replace fragments with newer ones, others keep old fragments



Audit Records

- a fundamental tool for intrusion detection
- two variants:
  - native audit records - provided by O/S
    - always available but may not be optimum
  - detection-specific audit records - IDS specific
    - additional overhead but specific to IDS task
    - often log individual elementary actions
    - e.g. may contain fields for: subject, action, object, exception-condition, resource-usage, time-stamp

Security Information and Event Management (SIEM)

- SIEM is a system combining Security Information Management (SIM) and Security Event Management (SEM).
- SEM deals with
  - Real-time monitoring
  - Correlation of events and threat intelligence
  - Notifications
  - Console views
- SIM deals with
  - Long-term storage
  - Analysis and reporting of log data

SIEM Features

* Source “ManageEngine” https://www.manageengine.com/products/eventlog/manageengine-siem-whitepaper.html

SIEM Workflow

SIEM Workflow

- Methods of collecting data from sources

- **Aggregation**: to gather data together as a whole in singular repository
- **Normalization**: to create consistent records by type and format

Normalization

- Original log format from source 1

  10:32, 12/3/2017, alsubaim, ad.corporate.com, error, failed login attempt

- Original log format from source 2

  12:45, 3/23/2017, malicious code detected, host1.corporate.com, alsubaim

- Normalized logs

  10:32, 12/3/2017, alsubaim, ad.corporate.com, failed login attempt

  12:45, 23/3/2017, alsubaim, host1.corporate.com, malicious code detected

SIEM Workflow

- Link events to identify attacks
- Event based:
  - a single event identifies an attack
- Rule based:
  - If X + Y + Z then do A
  - If X repeated 3 times within an hour, then do Y
- Anomaly based:
  - If the traffic on port X exceeds the standard deviation of historic traffic patterns, then there may be a problem

SIEM Workflow

- Severity

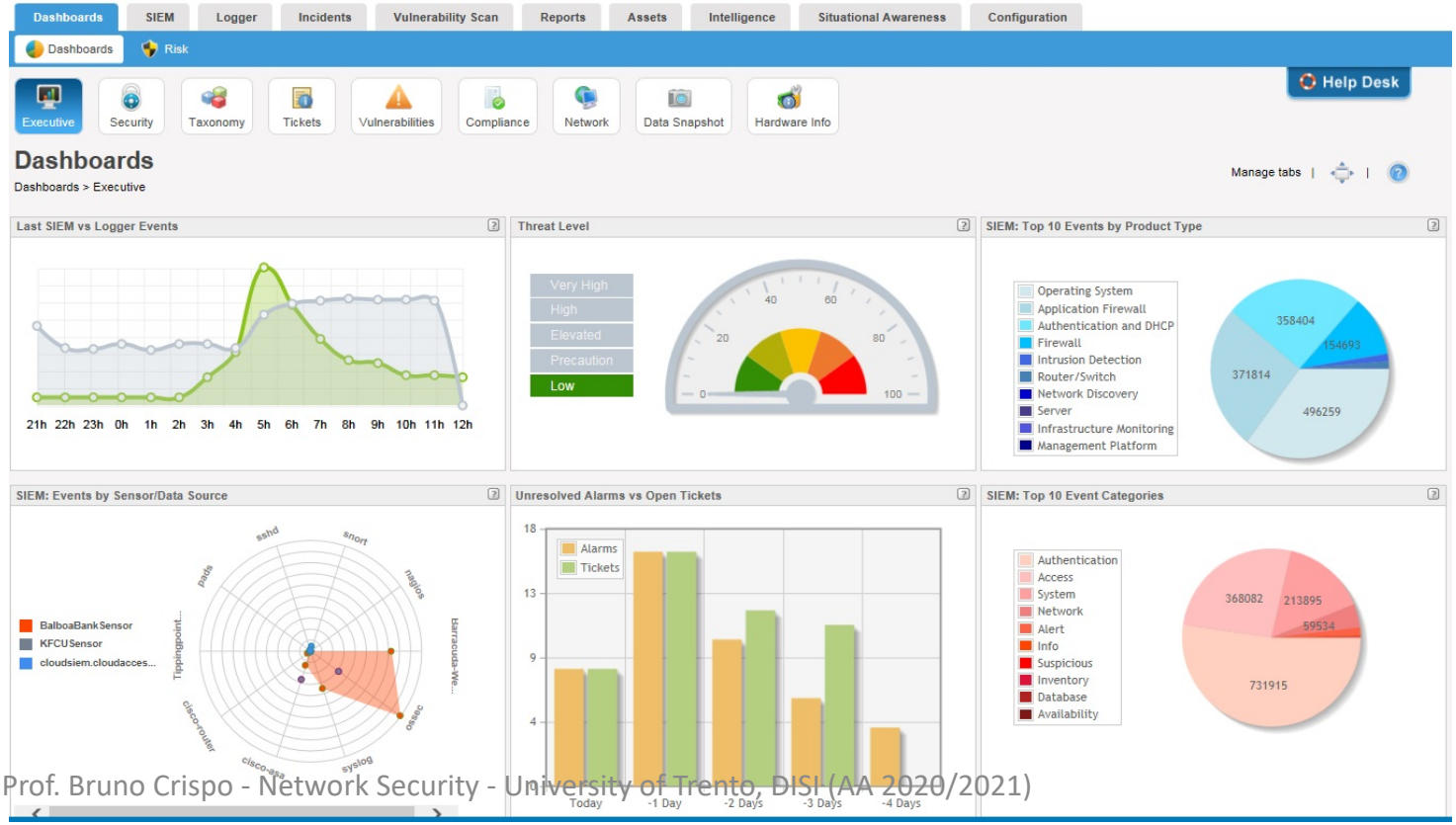
- Notification: upon identifying a threat, notifications are sent to the security administrators (SOC)
- Automated Response: the majority of SIEM tools can execute external scripts to react on identified threats. (Change to FW rules, issue a Remedy ticket)

UNIVERSITY OF TRENTO

Security Operations Center (SOC)

SIEM Workflow

Collect Data from Log Sources → Correlate Events → Alerts Security Incidents → **Generate Security and Compliance Reports** → Archive Logs for Forensic Analysis



SIEM Workflow

- Collected log data is stored for future forensic investigations.

- Not equivalent to Log Management Solutions

SIEM vs. LM

Functionality	Security Information and Event Management	Log Management
Log collection	Security related logs	All logs
Log pre-processing	Parsing, normalization, categorization, and enrichment	Indexing, parsing, or none
Log retention	Retain parsed and normalized data	Retain raw log data
Reporting	Security focused reporting	Broad use reporting
Analysis	Correlation, threat scoring, event prioritization	Full text analysis, tagging
Alerting and notification	Advanced security focused reporting	Simple alerting on all logs
Other features	Incident management, analyst workflow, context analysis, etc.	High scalability of collection and storage

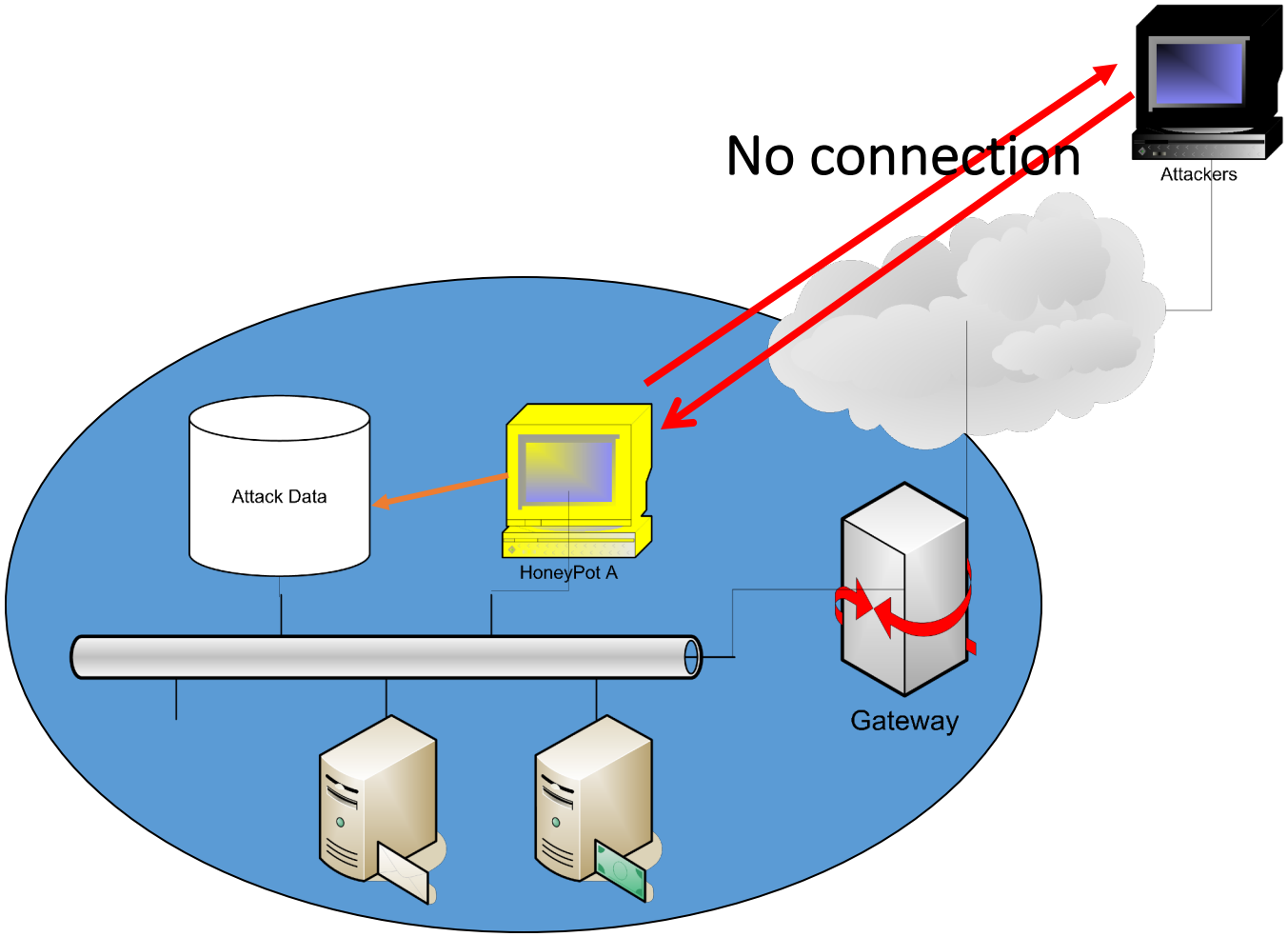
Honeypots

- are decoy systems
  - filled with fabricated info
  - instrumented with monitors / event loggers
  - divert and hold attacker to collect activity info
  - without exposing production systems
- initially were single systems
- more recently are/emulate entire networks

Physical V.S. Virtual Honeypots

- Two types
  - Physical
    - Real machines
    - Own IP Addresses
    - Often high-interactive
  - Virtual
    - Simulated by other machines that:
      - Respond to the traffic sent to the honeypots
      - May simulate a lot of (different) virtual honeypots at the same time

How do HPs work?



Production HPs: Protect the systems

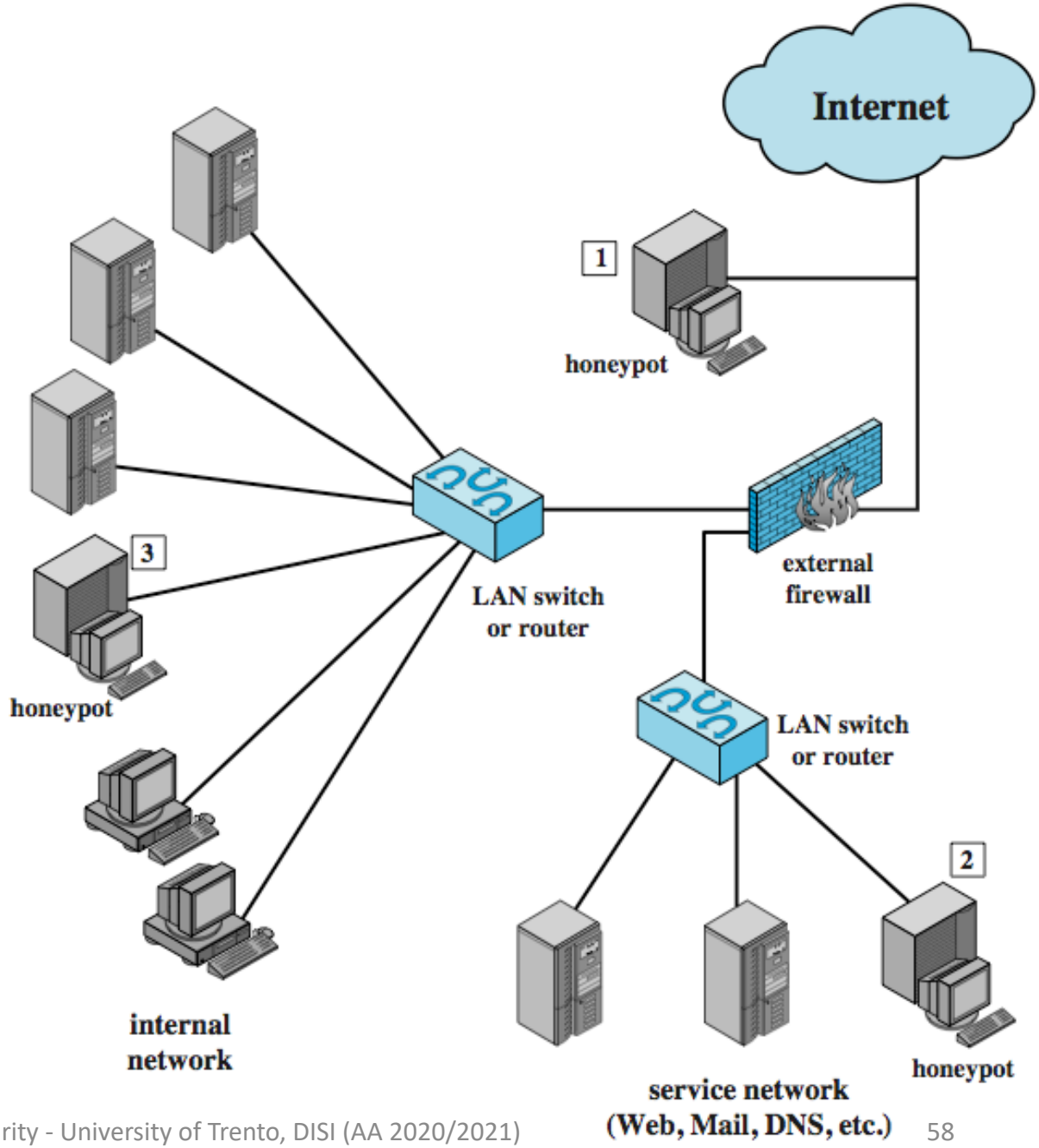
- Prevention
  - Keeping the bad guys out
  - not effective prevention mechanisms.
  - Deception, Deterence, Decoys do NOT work against automated attacks: worms, auto-rooters, mass-rooters
- Detection
  - Detecting the burglar when he breaks in.
  - Great work
- Response
  - Can easily be pulled offline
  - Little to no data pollution

Research HPs: gathering information

- Collect compact amounts of high value information
- Discover new Tools and Tactics
- Understand Motives, Behavior, and Organization
- Develop Analysis and Forensic Skills

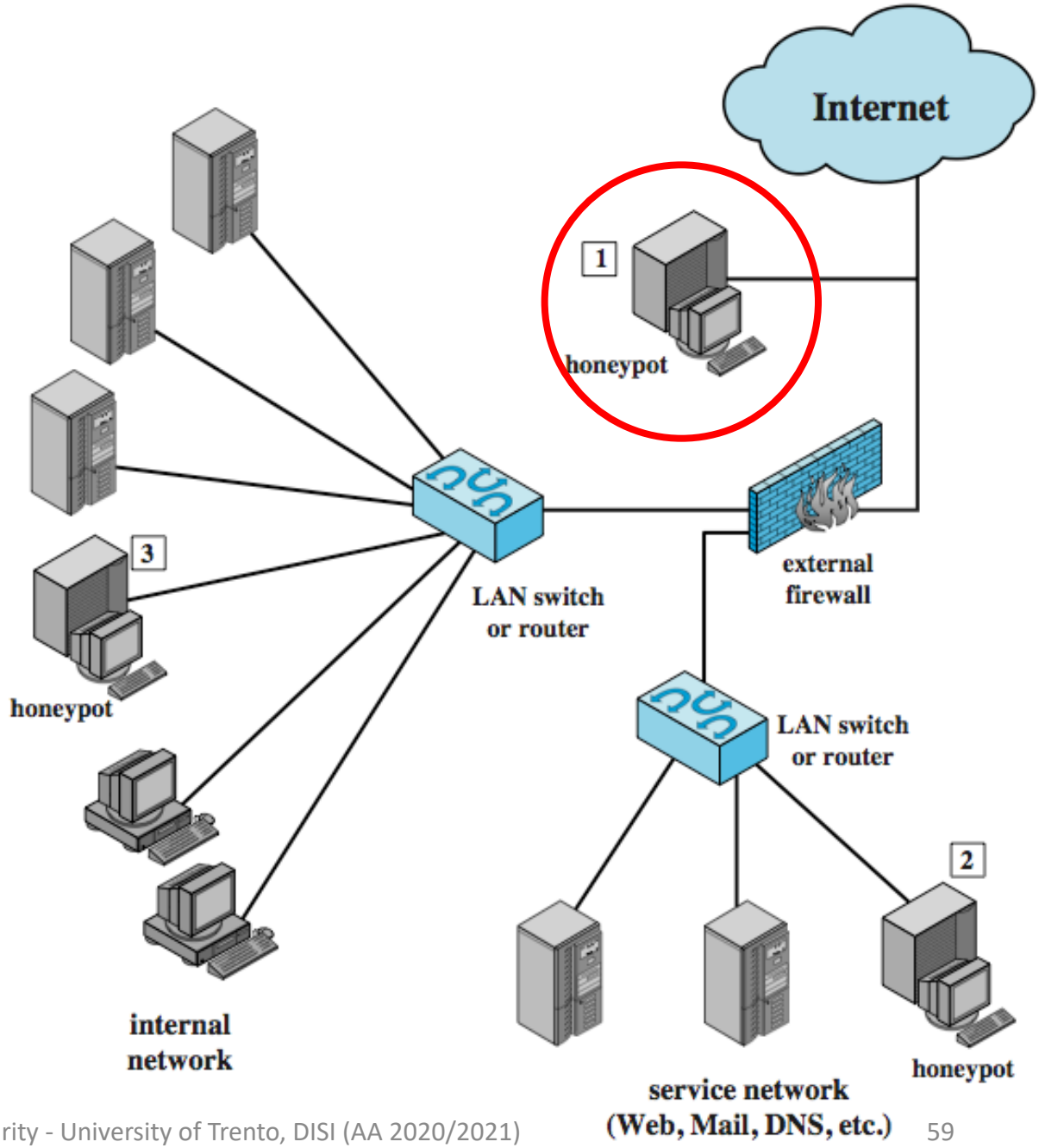
UNIVERSITY OF TRENTO

Honeypot Deployment



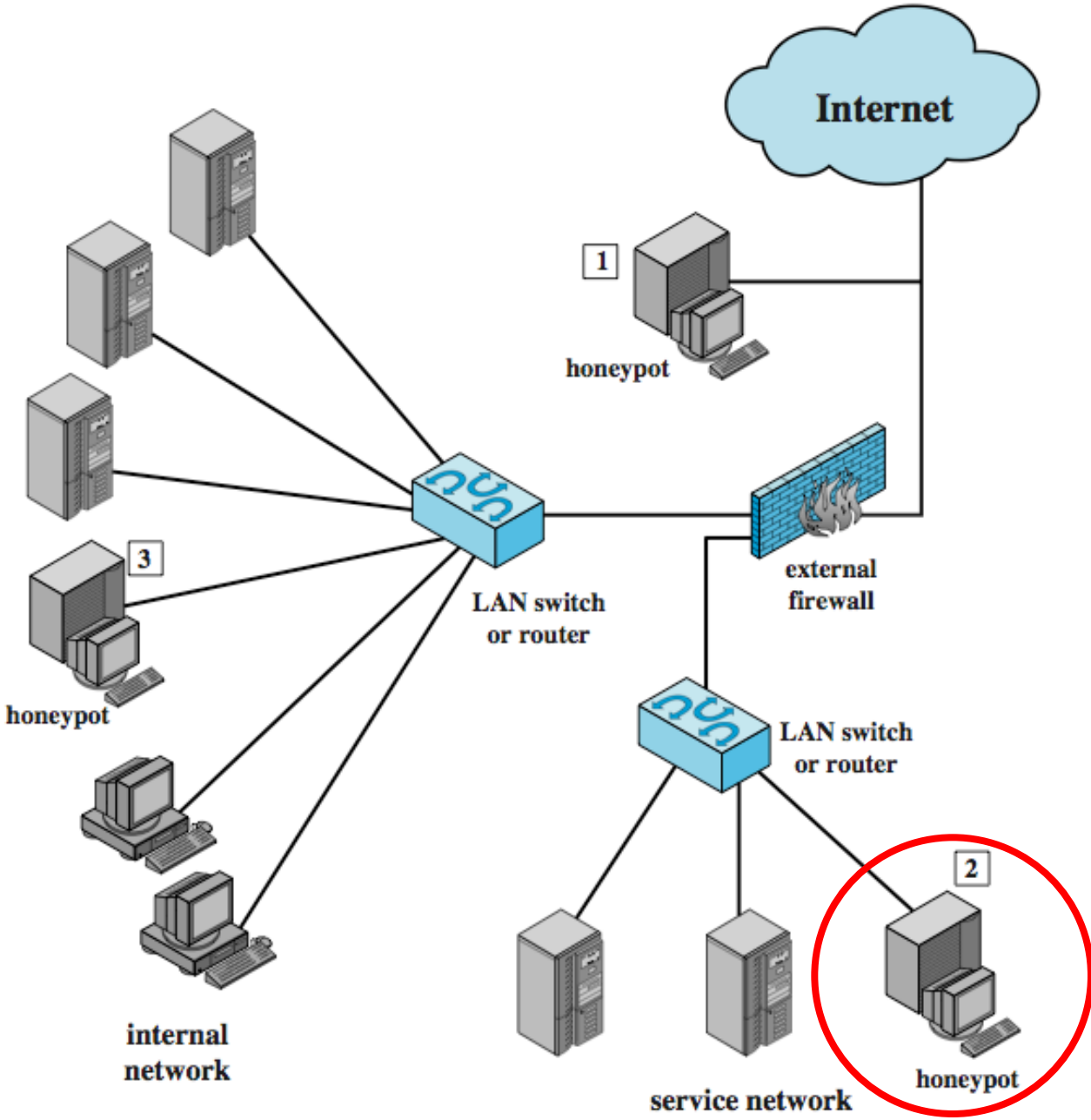
UNIVERSITY OF TRENTO

Honeypot Deployment



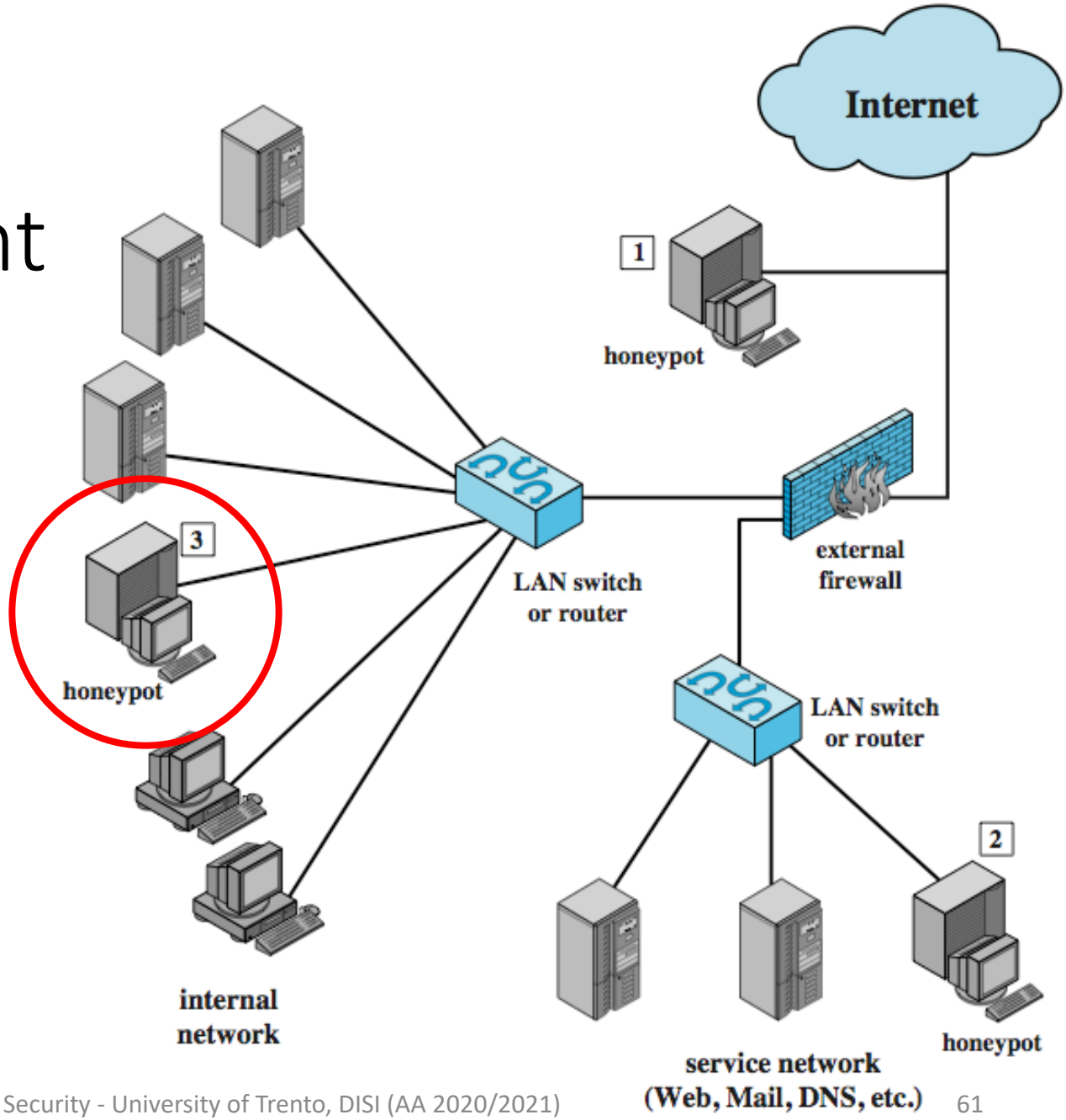
UNIVERSITY OF TRENTO

Honeypot Deployment



UNIVERSITY OF TRENTO

Honeypot Deployment



Suggested reading

- Wool, Avishai. "A quantitative study of firewall configuration errors." *Computer* 37.6 (2004): 62-67.
- Axelsson, Stefan. "The base-rate fallacy and the difficulty of intrusion detection." *ACM Transactions on Information and System Security (TISSEC)* 3.3 (2000): 186-205.
- [Siddharth 2005] https://community.broadcom.com/symantecenterprise/viewdocument/evading-nids-revisited?CommunityKey=1ecf5f55-9545-44d6-b0f4-4e4a7f5f5e68&tab=librarydocuments