Network Security

Intrusion detection

Marcus Bendtsen, Andrei Gurtov

Institutionen för Datavetenskap (IDA)

Avdelningen för Databas- och Informationsteknik (ADIT)

Intrusion Detection

- The goal of intrusion detection is to detect intrusions that have *occurred* or that are in the process of *occurring*.
  - Intrusion detection will do nothing to directly prevent intrusions, but may be helpful in attempting to understand them or mitigate their effects.
- Roughly divided into two parts:
  - Network Intrusion Detection Systems (NIDS)
    - Detect suspicious behaviour on the network
    - E.g., Snort, Bro
  - Host-based Intrusion Detection Systems (HIDS)
    - Detect suspicious activity on a single host
    - E.g., Windows Defender, F-Secure client

Intrusion Detection System (IDS)

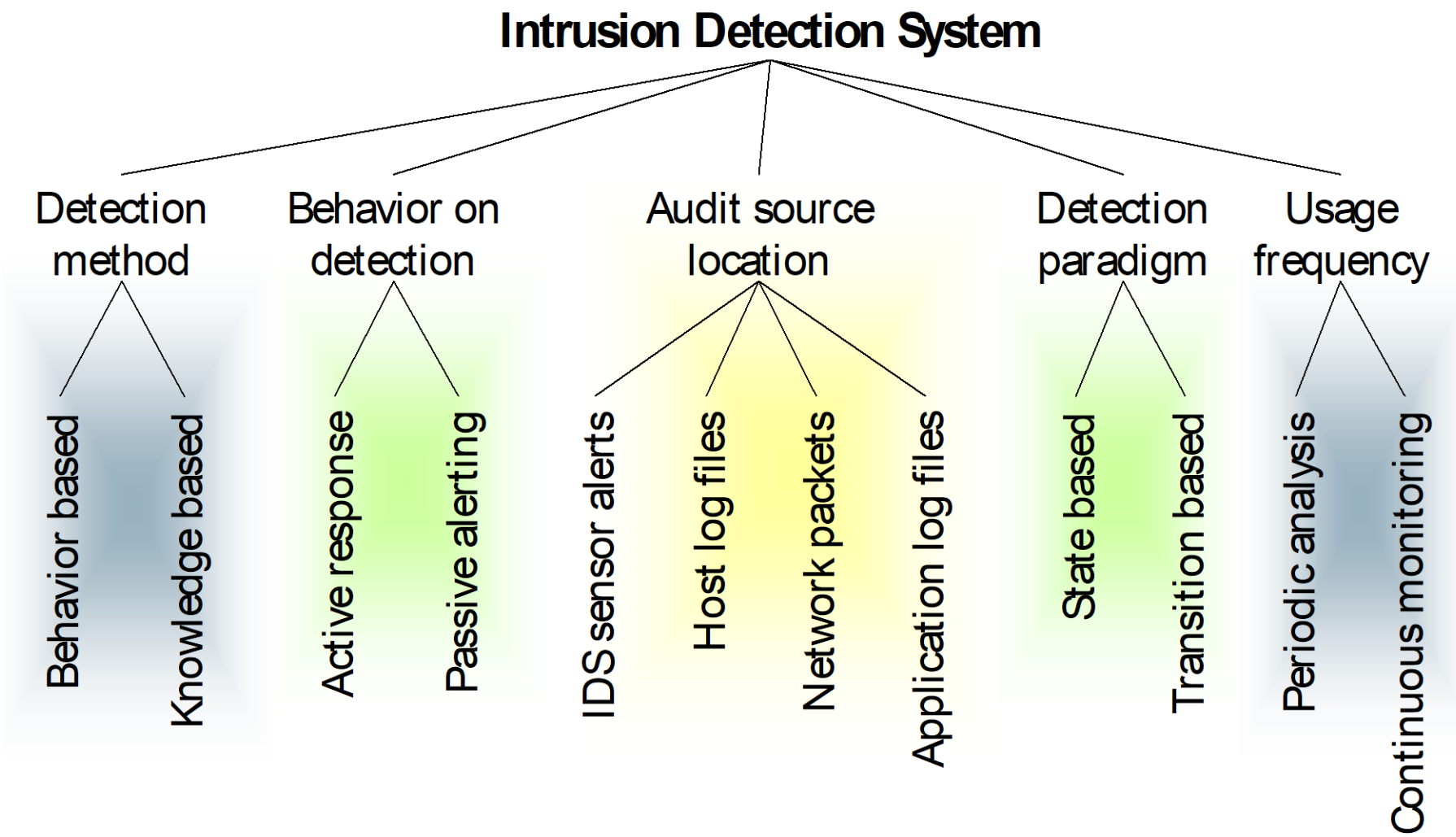
- All types of intrusion detection are really special cases of the more general concept of anomaly detection, which strives to detect anomalous behaviour in a system (which could be symptoms of misconfiguration, imminent equipment failure etc.)

- In general an IDS is a device that collects information or audits from any number of sources, analyses the information and determines whether there is a problem or not.

- The sheer volume of data available demands an automated approach, even if manual audits are of higher quality, it would be unfeasible to manually audit the systems continually.

IDS classification

- **Behaviour-based** systems use information about normal behaviour of a system in order to detect intrusions (which hopefully aren't normal).
- **Knowledge-based** systems use knowledge about intrusions (e.g. worm signature)
- **Behaviour on detection** separates systems that just raise an alert (passive) from systems that attempt to prevent the intrusion (active) or take other countermeasures.
- **The audit source** location is where the network-host separation comes in, advanced systems may be able to synthesize information from many sources.
- **Detection paradigm** separates systems that merely evaluate whether a system is secure or not (state based) or when a systems goes from secure to insecure (transition based).
- **Usage frequency** separates systems based on how often they run.


Intrusion Detection System
Detection method
Behavior based
Knowledge based
Behavior on detection
Active response
Passive alerting
Audit source location
IDS sensor alerts
Host log files
Network packets
Application log files
Detection paradigm
State based
Transition based
Usage frequency
Periodic analysis
Continuous monitoring

Example: Snort

- Snort is an open source NIDS, which can be classified as *passive alerting*, *network-packet-using*, *transition-based*, *continuous monitoring* system.

- It uses signatures from known attacks (*knowledge-based*) matches against network packets to detect attacks in real-time.

- Snort does include elements of behaviour-based IDS and active response.

Snort examples

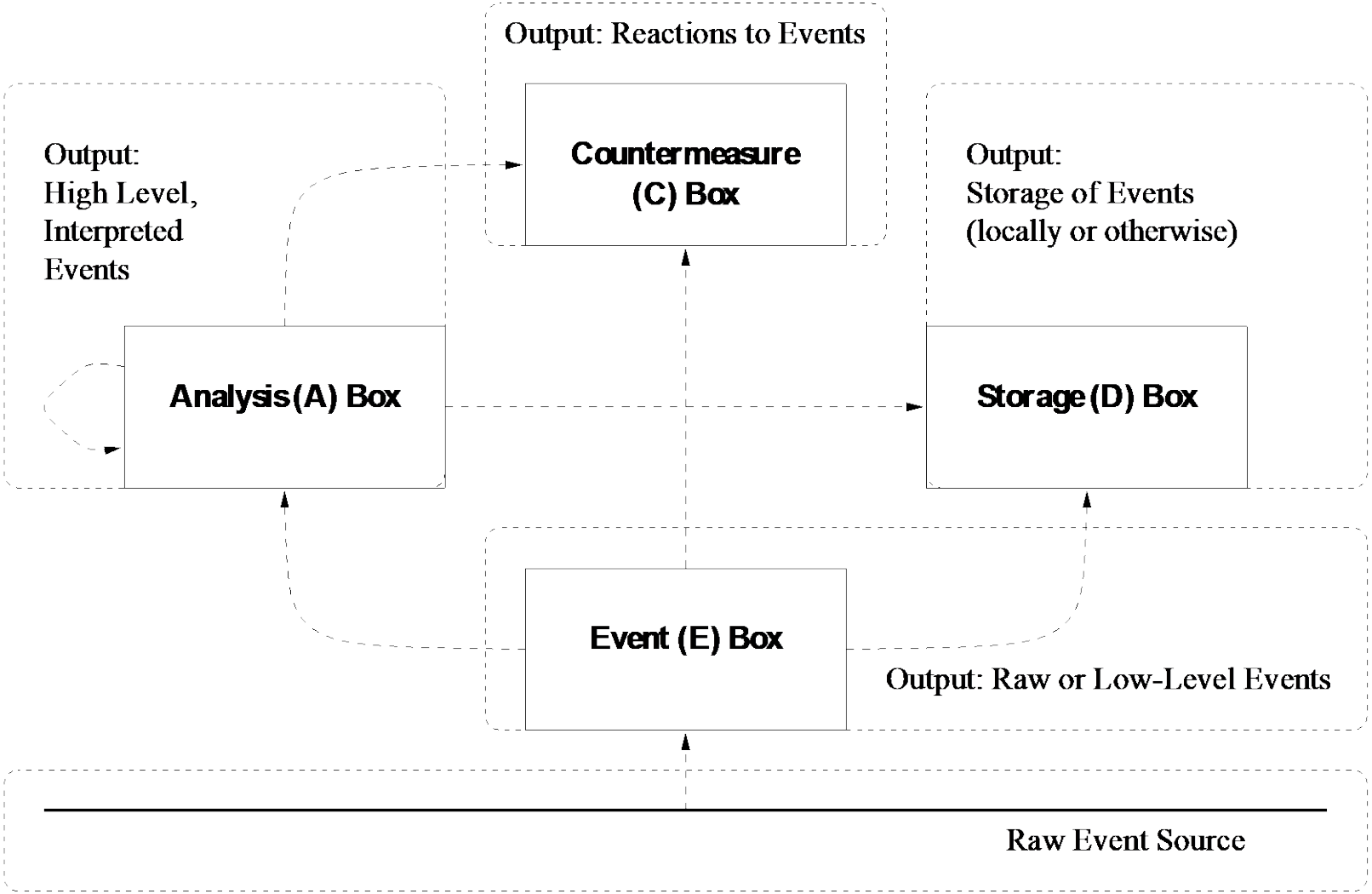
- Examples of rules in Snort
  - A database of rules that are matched against incoming traffic.

- Below is example that looks at external TCP connections that are trying to get a command line shell to a SQL server, this may indicate that a remote user has admin privileges, which is usually a bad thing.

```
alert tcp $EXTERNAL_NET any -> $SQL_SERVERS 445
     (msg:"MS-SQL xp_cmdshell program execution 445";
      flow:to_server,established;
      content:"x|00|p|00|_|00|c|00|m|00|d|00|s|00|h|00|e|00|l|00|l|00|";
      nocase;
      classtype:attempted-user; sid:1759; rev:5;)
```

Components

- Event generators (E-Boxes)
  - Provide information about events to the rest of the system.
- Analysis engines (A-boxes)
  - Analyses the information from event generators.
  - A lot of research goes into finding new ways of analysing data.
- Storage mechanisms (D-boxes)
  - Defines the means used to store security information.
  - Information gathered by E-boxes and analyses from A-boxes.
- Countermeasures (C-boxes)
  - Defines what to do if attack is identified.
  - Alarm, shut down, filter, etc.



Components of NIDS – T. Ptacek, T. Newsham – *Insertion, Evasion and Denial of Service: Eluding Network Intrusion Detection.*

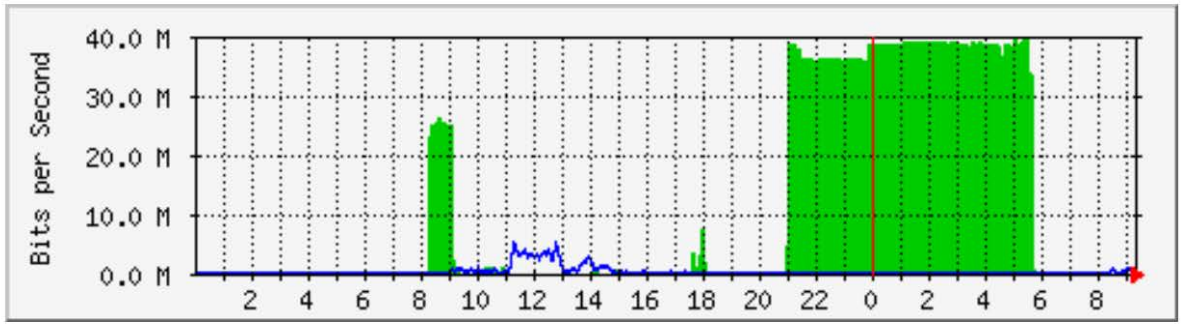
Points of vulnerability

- The purpose of an attack is *almost* always to hide another attack (I hit the NIDS first, then I can do whatever I want undetected).
- The components of NIDS can all be attacked.
  - Hitting the **E-boxes** effectively disables NIDS, as it has no input to use for detection.
  - Hitting the **A-boxes** disables all analysis, so even if the NIDS has lots of data to work on, nothing can be done.
  - Hitting the **D-boxes** can result in data not being stored for analysis, or maybe manipulated so that *evil* data is made to look safe.
    - Also, the NIDS saves a lot of information about hosts on the network. This needs to be stored somewhere. Getting this information can be very useful for the attacker.
  - Hitting the **C-boxes** results in no action being taken if an attack is found. This may be enough for the attacker, as it may result in no logs being written and all services working as normal.

NIDS challenges

- NIDS are not fool-proof, there are significant challenges to overcome in implementation.
  - **False positives** – A NIDS facing heavy traffic may raise alarms for legitimate traffic.
    - A NIDS looking for strings that are commonly used to get to backdoors may detect the same strings in legitimate traffic.
    - A NIDS may detect a SYN flood where the reality was a sudden surge in traffic, due to increase popularity of a website.
    - NIDS needs to be carefully tuned, as introducing *false negatives* when attempting to reduce false positives is also undesirable.

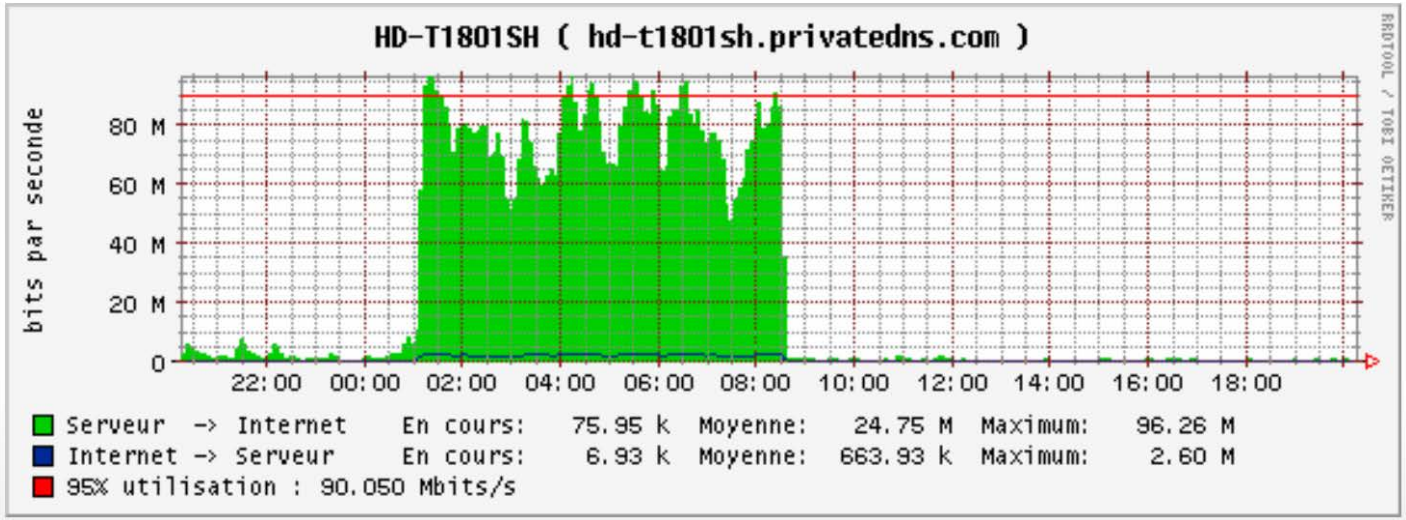
DoS or not?



Slashdot Effect?

Night backup?

Intruder stealing data?



NIDS challenges

- NIDS are not fool-proof, there are significant challenges to overcome in implementation.
  - **Performance** – NIDS that scans high traffic volumes require high performance hardware, networking and software to work.
    - NIDS that misses traffic regularly may be unable to detect anomalies.
    - A NIDS needs to defragment all datagrams that it sees, and needs to track all TCP connections to all hosts in the network, at least long enough to establish that the connection is safe.
  - **Security features** – IPSec, VPN tunnels, TSL/SSL and application-layer encryption are all designed to protect sensitive information from prying eyes; but they also prevent NIDS from doing its job.
    - Difficult to circumvent, major issue for NIDS.

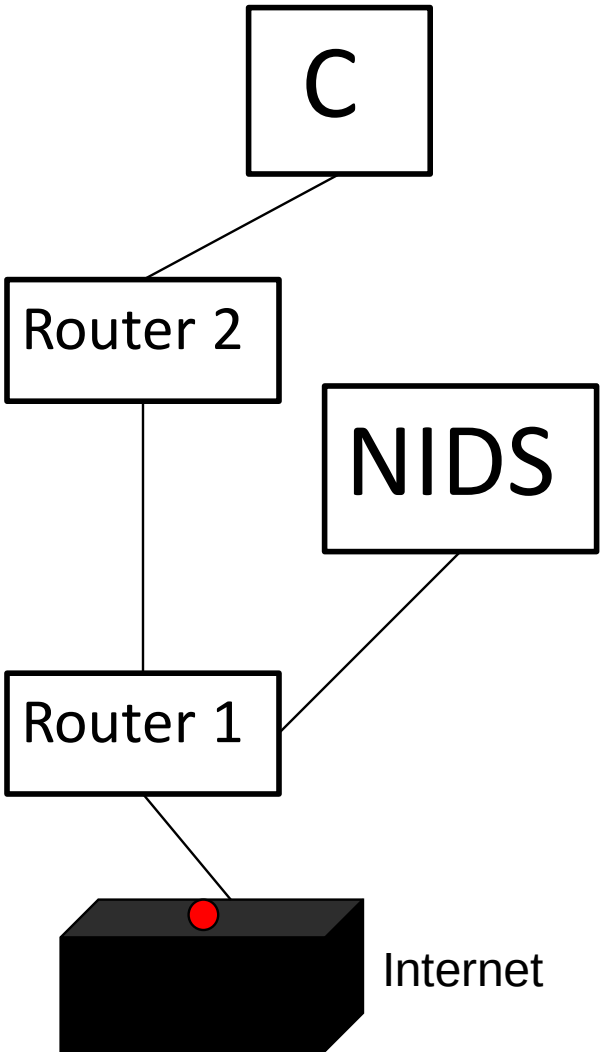
NIDS challenges

- NIDS are not fool-proof, there are significant challenges to overcome in implementation.
  - **Desynchronization –** NIDS needs to mimic the processing of the end system.
    - In order for NIDS to understand the consequence of some particular network traffic, NIDS needs to know what state the end system is in.
    - The effect of the network traffic should be understood by NIDS.
    - If the end system behaves differently to some traffic than NIDS simulates, then the systems are desynchronized, and no further accurate detection can be done.

Example of attack on E-box

- I know that host **C** has a vulnerability. It implies that if I send the word HACK, it will shutdown.
- I also know that **NIDS** is aware that this vulnerability may exist, and I do not want to get caught.
- I have good knowledge about the network layout, and I know that **NIDS** is sitting on one network segment prior to **C.** (May have scanned to get this information).

- I will send the work HQACK to **C**, but I will make sure that the datagram containing Q will have a TTL set such that **NIDS** will receive it, but **Router 2** will decrement it to 0 and respond with ICMP.
- H, A, C, K however does make it through **Router 2** as they have longer TTLs.
- **NIDS** sees HQACK, **C** sees HACK



Other exploits

IP fragmentation

- IP packets can be broken up into smaller packets for transit. End-systems need to reassemble fragments in order to get the original IP packet.
- In order for NIDS to see the entire IP packet it needs to collect all the fragments.
  - What if I send a *never-ending* stream of fragments without any fragment marked as *final*? Eventually I will use all the NIDS resources and it will be unable to operate properly. (*Dos*)
- NIDS also needs to reassemble the fragments, and has to do so exactly the same way as the client it is monitoring traffic for.
  - If two fragments for some reason overlap (they can be of different size and come in different order), NIDS needs to handle this ***exactly*** the same way as the client will.
  - In some situations old data is replaced, in some situations new data is discarded. Turns out that operating systems differ in their behaviour.

Other exploits

Abusing reactive ID systems

- Some NIDS do not only create logs and warnings, but actually take action (a more reactive C-box).
- This can be exploited to effectively make the NIDS perform a DoS attack on the network it actually wants to protect.

- Imagine that I attack the system with a SYN flood, and to not be detected I fake the source IP.
- The NIDS sees the SYN flood attack and takes action to terminate all connections from this IP.
- I do this for IP address blocks allocated to in Sweden, and suddenly connections from Sweden are disallowed.

NIDS lessons

- NIDS will never be 100% reliable, so a NIDS should never be the only solution. Host-based IDS and other monitoring systems should be used.

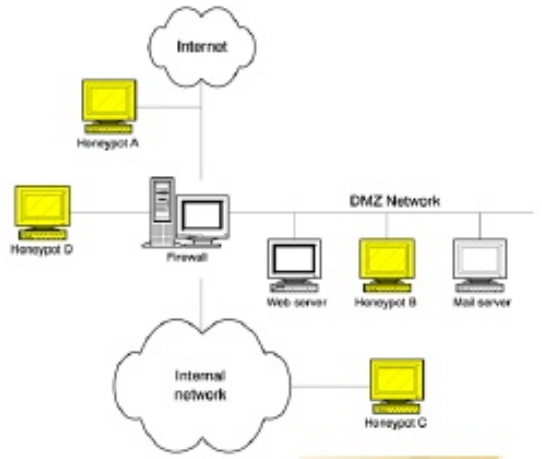
- **Trade offs – Common in security**
  - Trade offs – Common in security – The more diligent the NIDS is to keep synchronisation, the more resources are needed.
  - If you add security such as IPsec, SSL/TSL, then NIDS will suffer.
  - Active NIDS can be really useful to stop attacks, but can be exploited by attacker.

- NIDS also teaches us that it is important to know how protocols work, and to also verify that they work as expected.

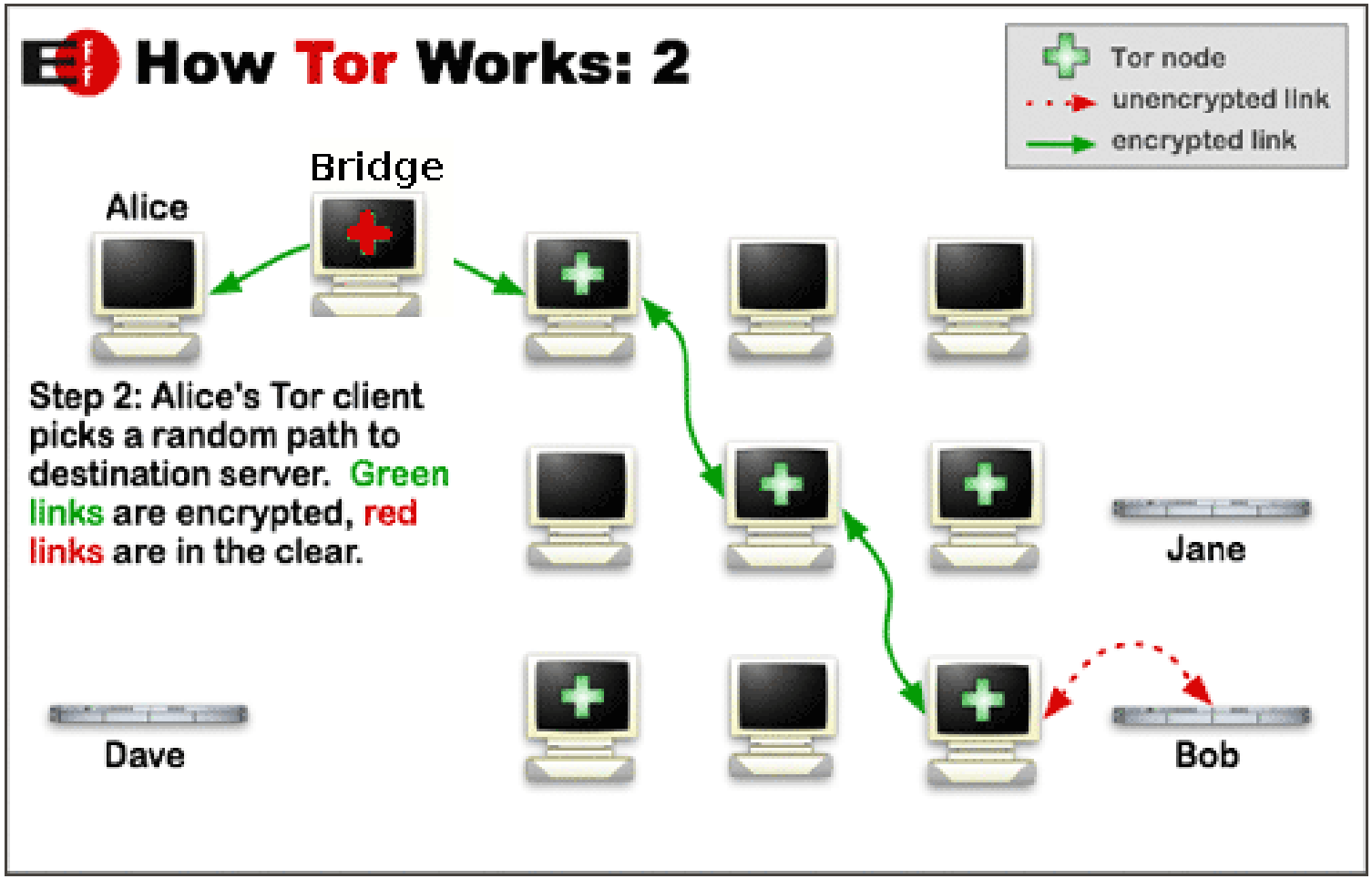
Honeypots with Known Vulnerabilities

Honeypot can be placed:

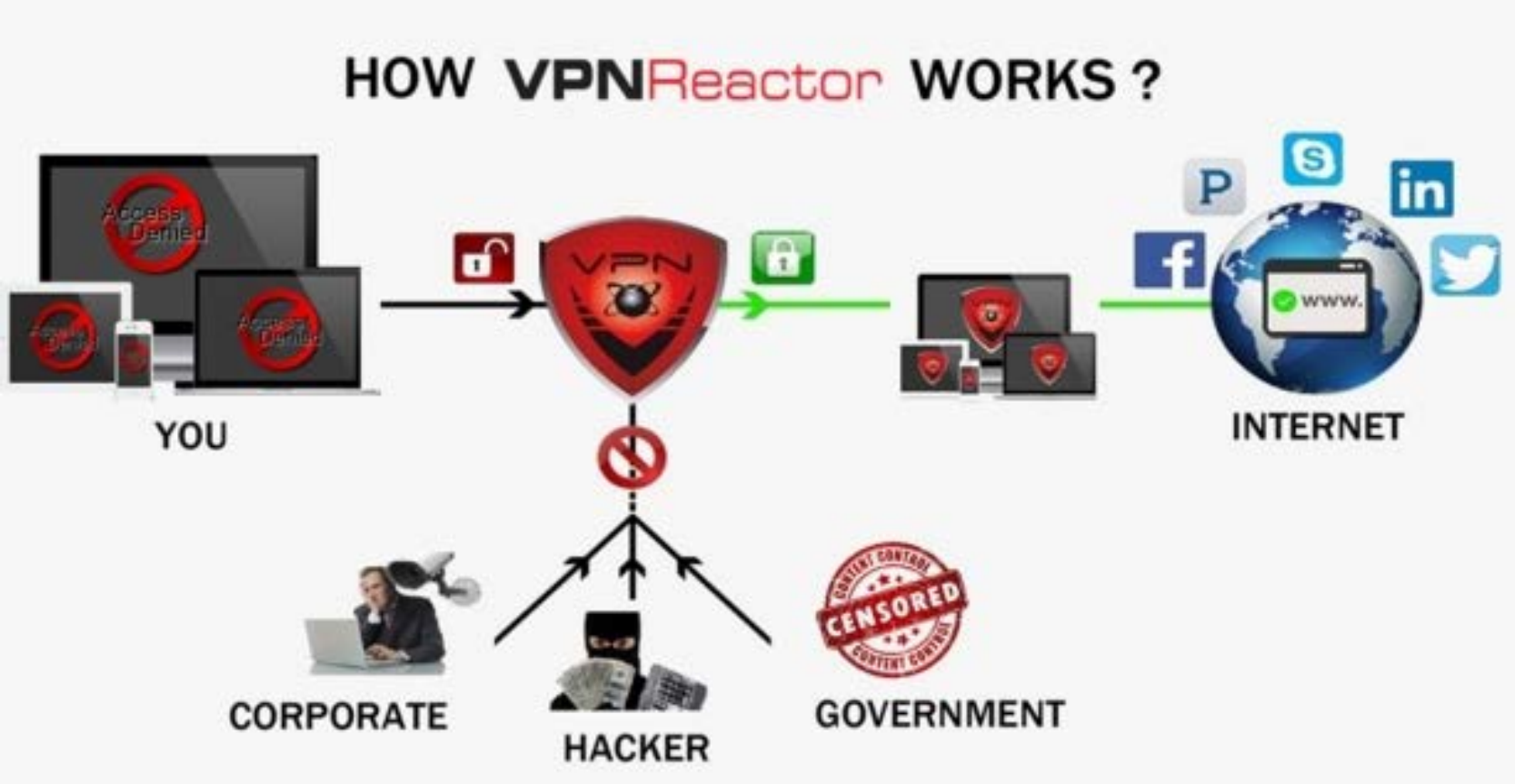
- *In front of the firewall (Internet)*
- *DMZ (DeMilitarized Zone)*
- *Behind the firewall (intranet)*



Tor Onion Routing/DarkNet

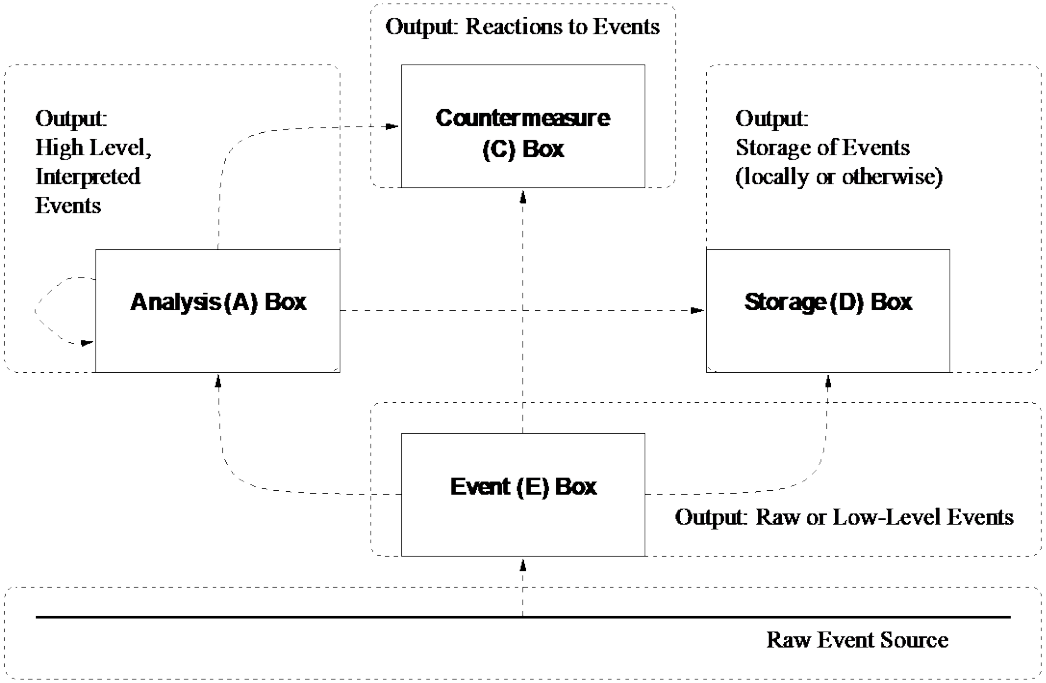


Virtual Private Network Anonymizer



Summary

- IDS is crucial to detect and react to attacks.
- It gives us valuable information for future security improvements.
- Many trade-offs needs to be made, and NIDS is not enough on its own.
- All four “boxes” can be attacked:
  - Can lead to attacks not being noticed.
  - Can lead to the NIDS itself creating DoS attacks etc.
  - Can lead to network system configuration leaks.



Components of NIDS – T. Ptacek, T. Newsham – *Insertion, Evasion and Denial of Service: Eluding Network Intrusion Detection.*

Linköpings universitet

expanding reality

www.liu.se