Intrusion Detection Systems

Lesson #4: Honey{pots,nets,walls}

Matthijs Koot
(koot@uva.nl)

Faculteit van Natuurwetenschappen, Wiskunde en Informatica
Universiteit van Amsterdam

2007-04-12 / SNE-IDS college '06-'07

Outline

Matthijs Koot
(koot@uva.nl)

Definitions: ‘honeypot’.

Matthijs Koot
(koot@uva.nl)

Definition

A honeypot is a sacrificial asset, either virtual or physical, whose purposes are to gather information on (and warn of) attacker behavior and to decoy attackers from real assets.

Definitions: ‘Honeynet’ and ‘HoneyWall’.

Definition

A honeynet is a network of honeypots.

Definition

A honeywall is a honeypot or honeynet that is placed in-line between two networks, or between a network and a host, to uni- or bidirectionally capture, control and analyze attacks.

Definition

A honeytoken is a honeypot which is not a computer.

Warning.

WARNING

In real life, ‘Honeynet" and “HoneyWall" are used ambiguously to refer to both their CONCEPTS, as well as their prevalent IMPLEMENTATION (think ‘DNS’ versus ‘bind’). This also explains any inconsistencies in (my) use of CaPiTaLiZaTiOn.

Purpose of a honeypot.

The two main purposes of a honeypot:

- Research
  - Know your enemy!
  - Reveal blackhat tactics, techniques, tools
  - Reveal motives/intentions?
  - Mostly used by universities, governments, ISPs
- Protection
  - Deceiving the wiley cracker
  - Integrated within enterprise security architecture

Tactics behind a honeypot.

In its defensive form, a honeypot is designed on deception and intimidation (Fred Cohen, 2001):

- Concealment
- Camouflage
- False/planted information (honeytokens!)
- Feints, lies, et cetera
  - E.g. false claims that a facility if being watched by law enforcement authorities

Functional requirements of a honeypot.

Functional requirements of a honeypot include:

- Data capture
- Data analysis
- Data control

Matthijs Koot
(koot@uva.nl)

History of honeypots.

- 1990: real systems
  - Deploy unpatched systems in default config on unprotected network (making them ‘low-hanging fruit’)
  - Easy to deploy
  - High-interaction, high-risk
  - Nice reading: “Cuckoo’s Egg” by Clifford Stoll
- 1998: network service simulation
  - HoneyD, CyberCop Sting, Deception Toolkit, KFSensor
  - Easy to deploy
  - Low-interaction, low-risk
- 1999-now: virtual systems
  - Honeynet (next slide), Symantec Decoy Server
  - Difficult to deploy
  - Mid/high-interaction, mid/high-risk

Matthijs Koot
(koot@uva.nl)

History of The Honeynet Project.

History of The Honeynet Project

- 1999: Lance Spitzer (Sun) founds Honeynet project
- 1999-2001, GenI: PoC, L3+ (modified IP-headers)
- 2001-2003, GenII: GenI + bridging (no TTL, harder to detect)
- 2003-2004, GenIII: GenII + blocking (HoneyWall)
- 2004-current: 'GenIV' refers to next-gen analysis capabilities

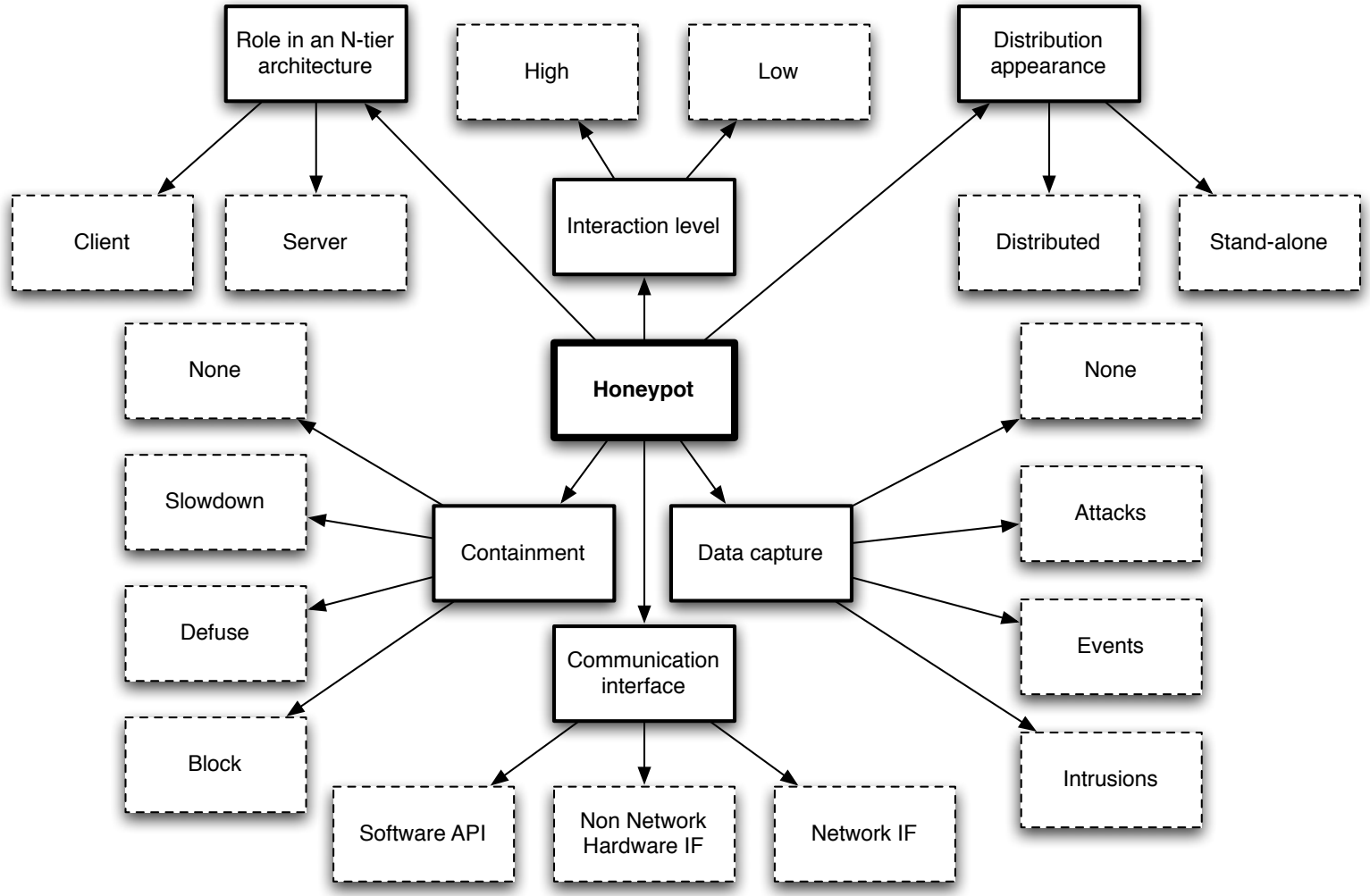
Also known for Scan of the Month (SotM) challenges (which alas appear to be dormant since 2005)

Taxonomy of honeypots.

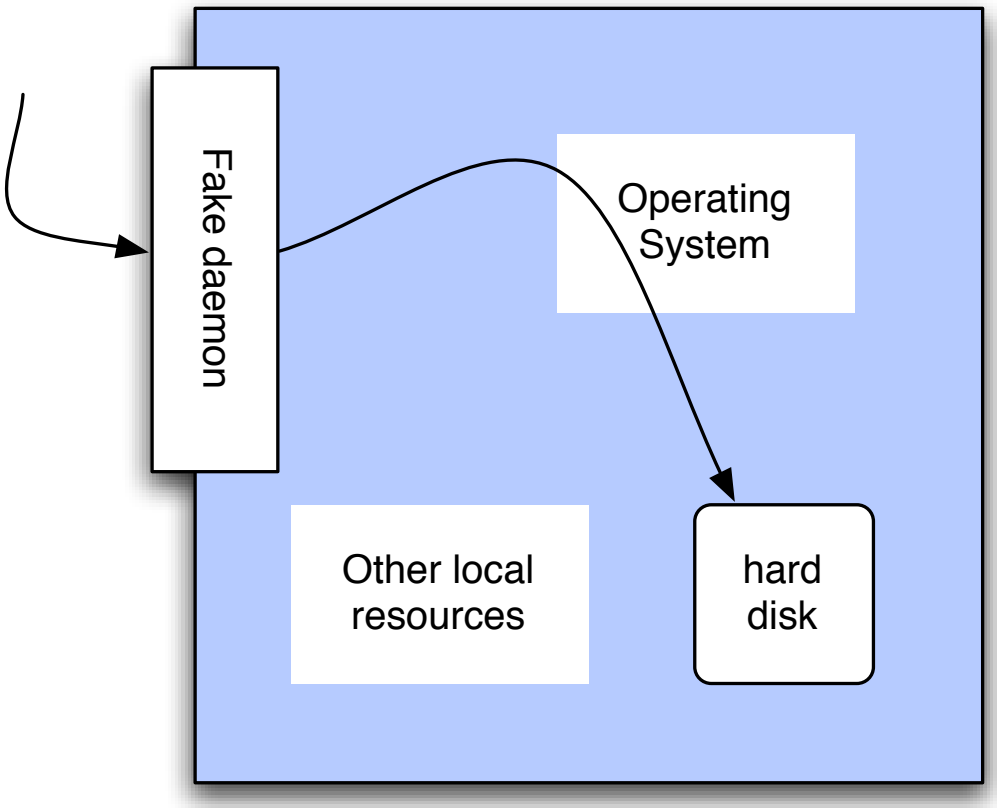
As proposed by Seifert, Welch, Komisarczuk in 2006, honeypots can be differentiated on...

- Level of interactivity (will be discussed shortly)
- Data capture (attacks, events, intrusions, ...)
- Containment (aka ‘data control’)
- Distribution appearance
- Role in N-tier architecture
- Communication interface (API, NIC, ...)

Taxonomy of honeypots.

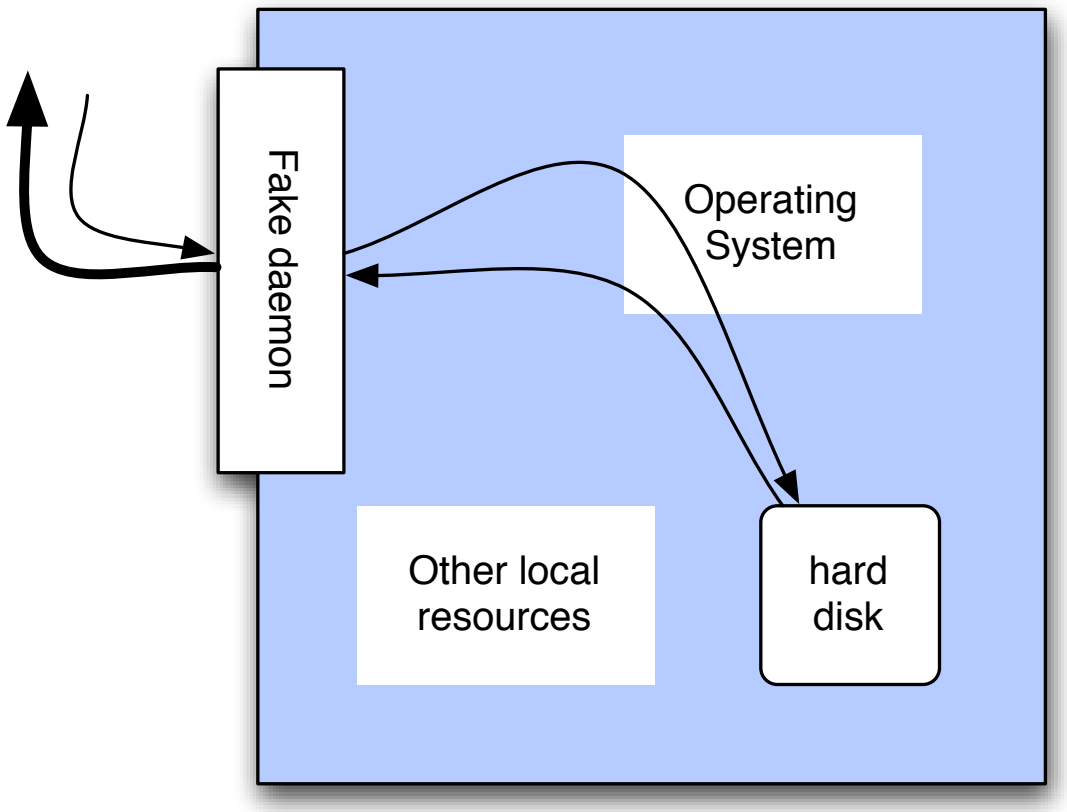


Level of interactivity: low.



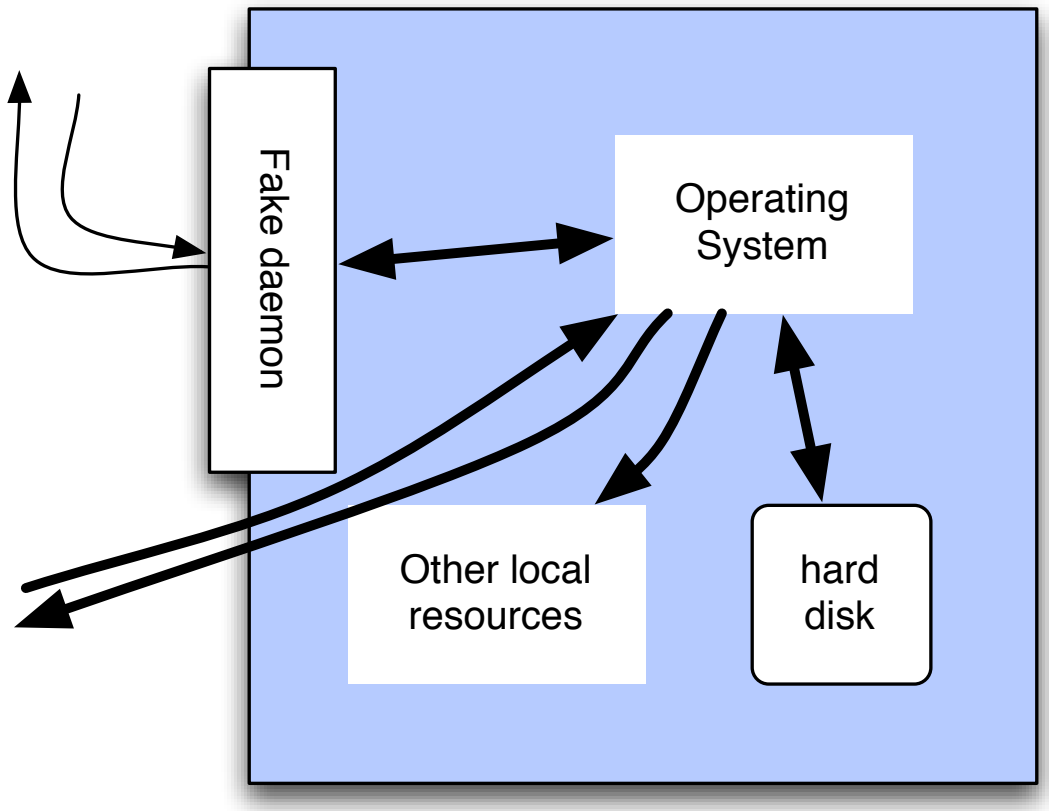
Source: R. Baumann, C. Plattern. Honeypots, diploma thesis. Feb. 2002 (reconstructed for OS3/SNE)

Level of interactivity: mid.



Source: R. Baumann, C. Plattern. Honeypots, diploma thesis. Feb. 2002 (reconstructed for OS3/SNE)

Level of interactivity: high.



Source: R. Baumann, C. Plattern. Honeypots, diploma thesis. Feb. 2002 (reconstructed for OS3/SNE)

Outline

Matthijs Koot
(koot@uva.nl)

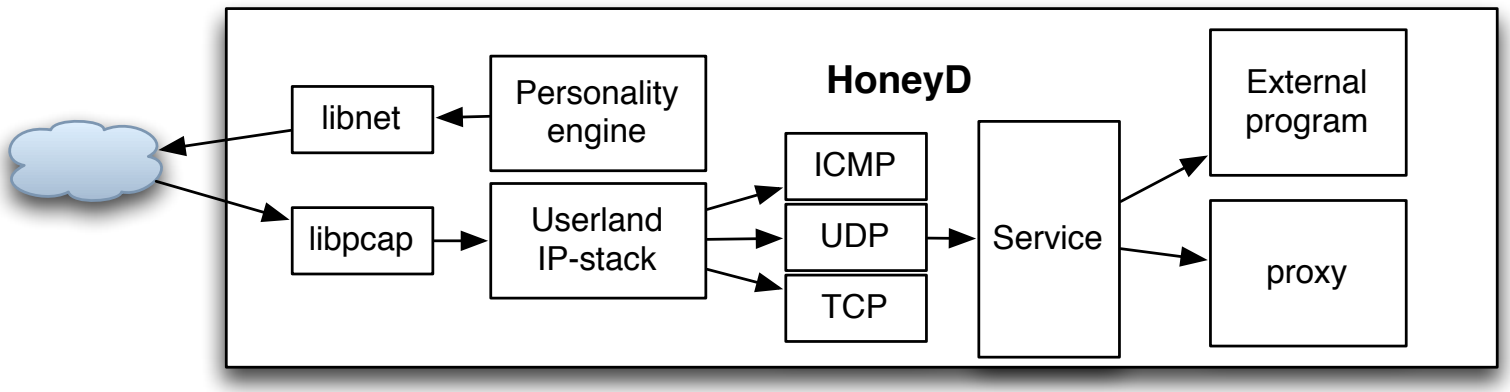
HoneyD.

HoneyD

- HoneyD is an engine for running virtual IP-stacks in parallel
- Mid-interaction network service simulator
  - Simulates SMTP, FTP, HTTP, ...
  - Easily extendible through customizable scripts
- First release in 1999, currently maintained by Niels Provos
- TCP/IP fingerprint spoofing through ‘personalities’
  - Impersonate Win32 on your favorite UNIX flavor (which should be MINIX), fooling nmap and xprobe
  - Fake WinSize, DF, ToS, ISN, ...
  - Fake packet loss, TTL, latency!

HoneyD.

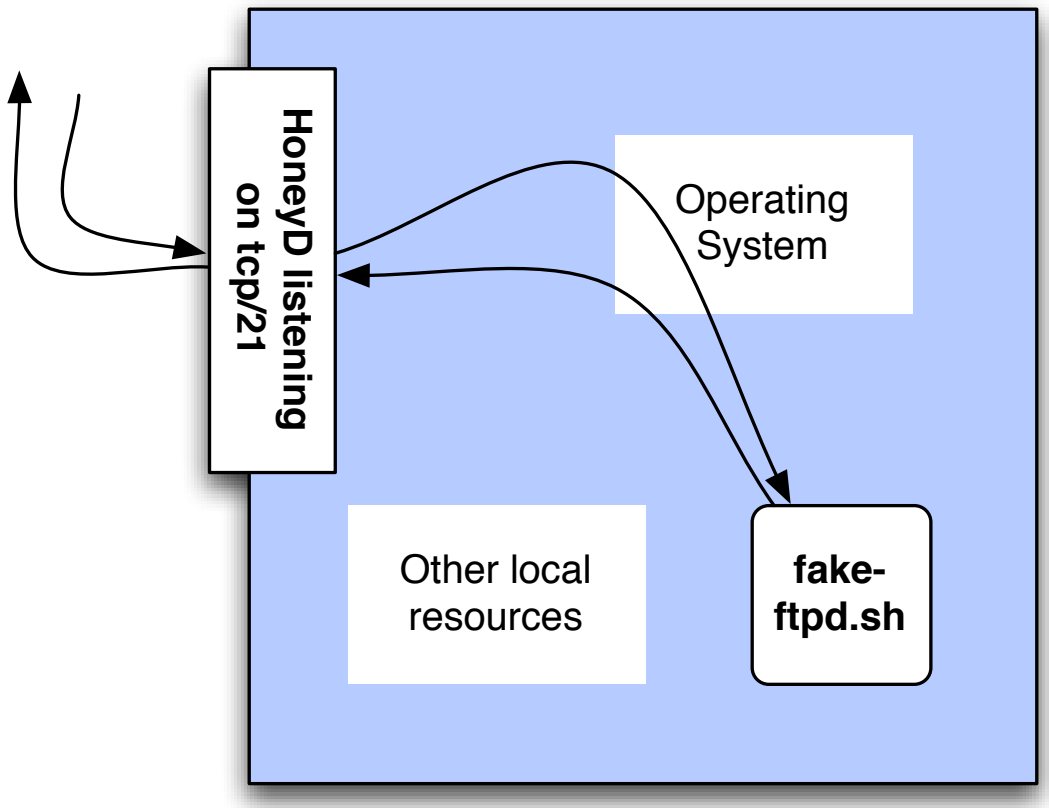
HoneyD architecture



Source: http://md.hudora.de/presentations/2005-bh-honeypots-03-honeyd.pdf (reconstructed for OS3/SNE)

HoneyD.

Applying the mid-interaction model to HoneyD: HoneyD servicing incoming requests on port TCP/21 by executing `fake-ftp.sh`.



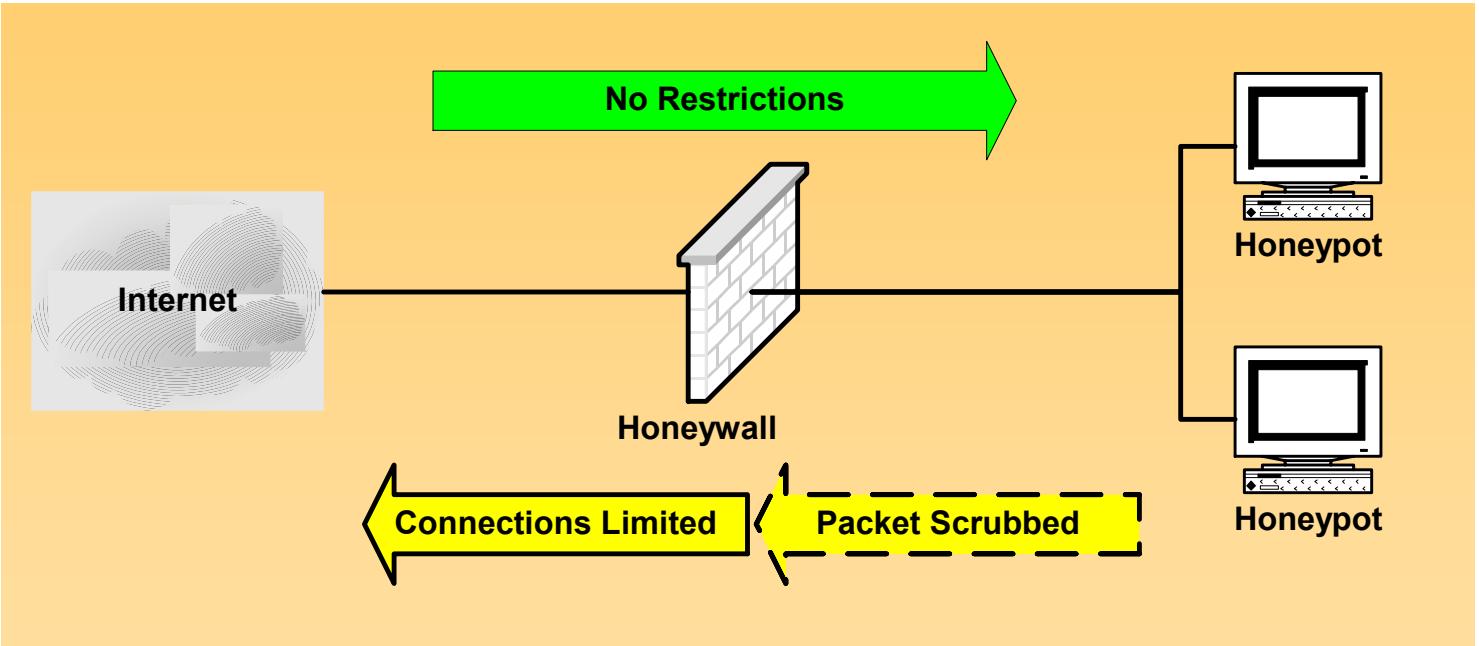
Outline

Matthijs Koot
(koot@uva.nl)

Honeynet, HoneyWall.

Matthijs Koot
(koot@uva.nl)

The basic idea of a Honeynet/HoneyWall:



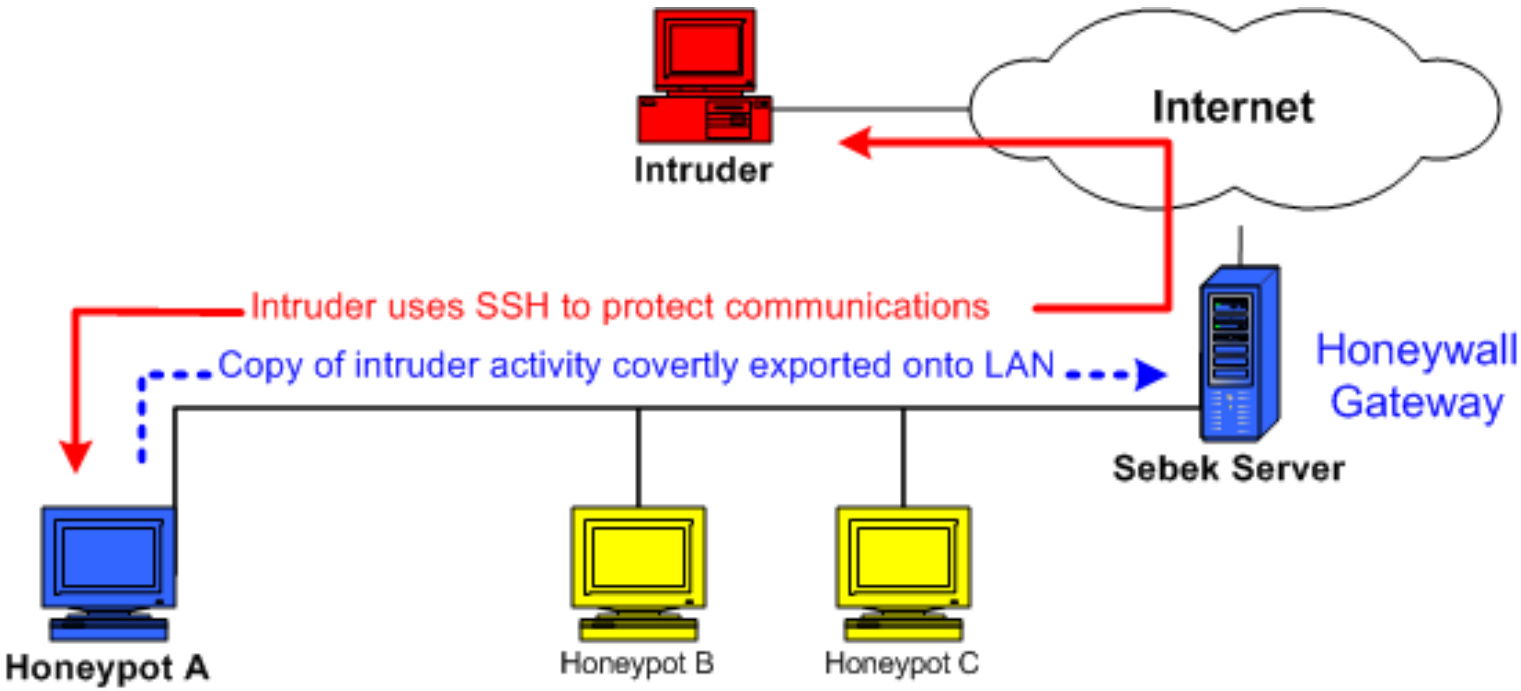
Sebek.

Sebek: spying on your intruder

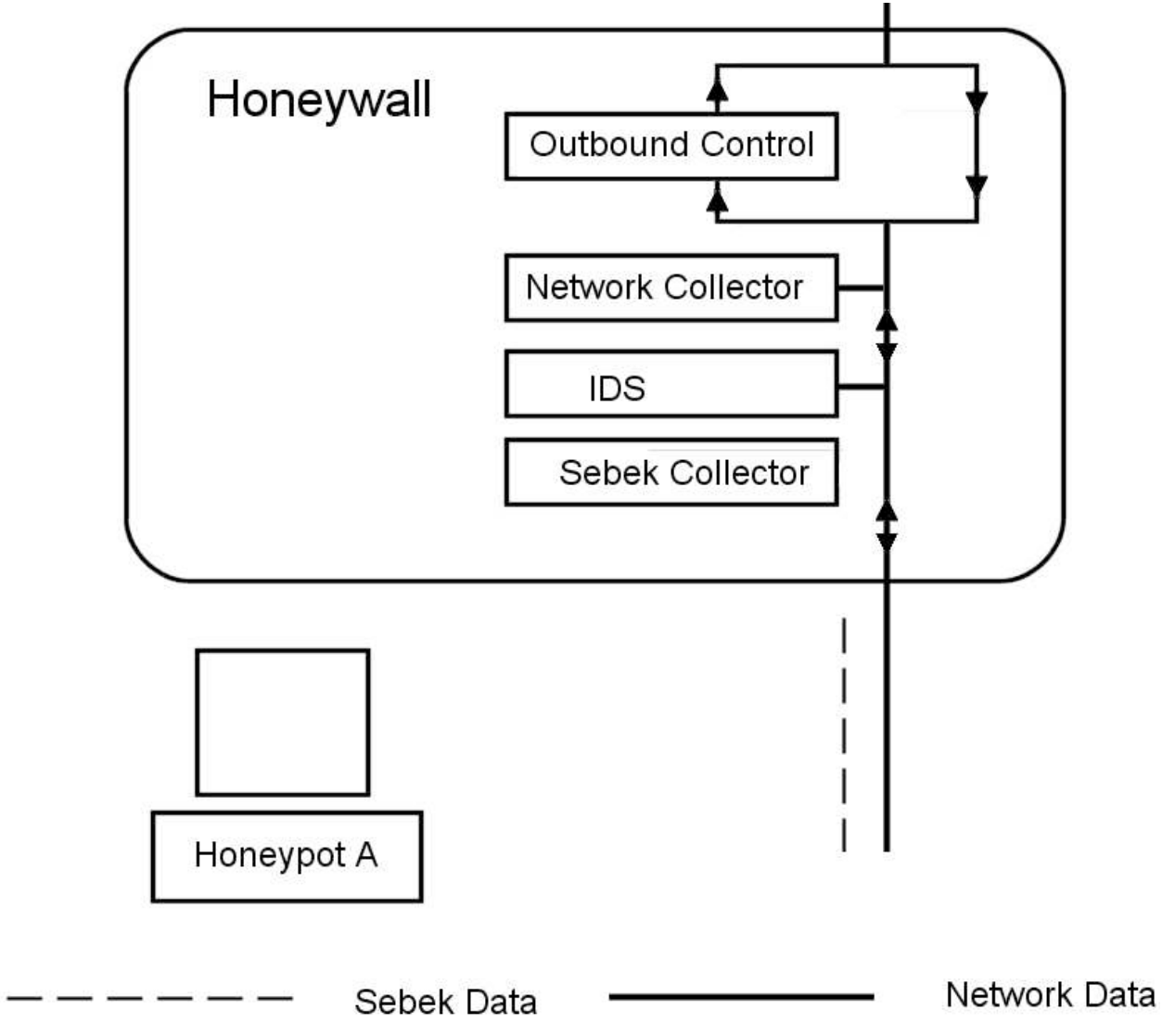
- From Honeynet.org: “Sebek is a tool designed for data capture, it attempts to capture most of the attackers activity on the honeypot, without the attacker knowing it (hopefully), then sends the recoverd data to a central logging system."
- Recover keystrokes, uploaded files, passwords, IRC chats, even if they’re encrypted by SSH, IPSec or SSL.

Sebek.

Matthijs Koot
(koot@uva.nl)

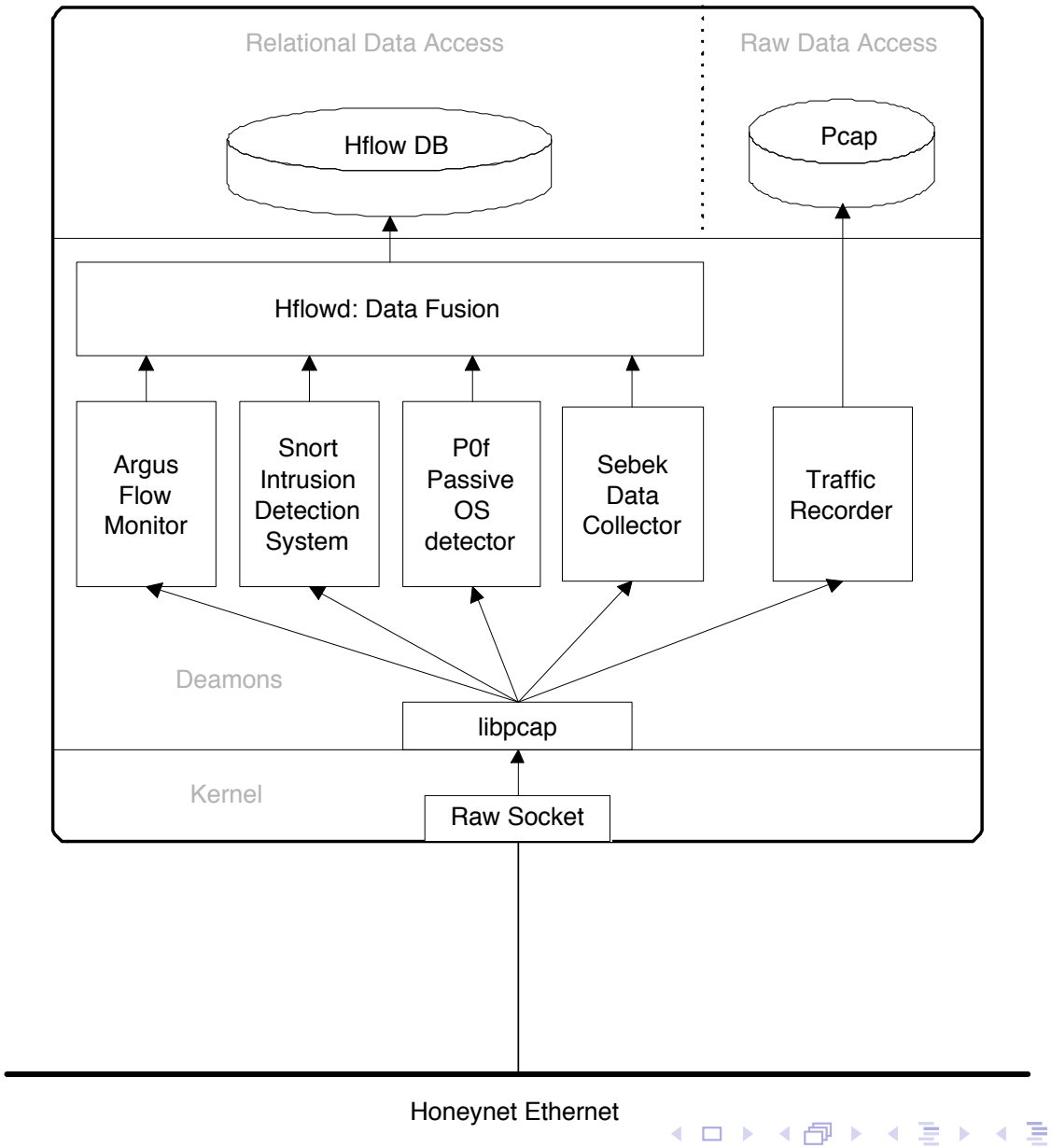


Sebek in GenII honeynet.



Matthijs Koot
(koot@uva.nl)

Hflowd data fusion.



Outline

Matthijs Koot
(koot@uva.nl)

MWCollect: Nepenthes, HoneyTrap and HoneyBow.

Matthijs Koot
(koot@uva.nl)

MWCollect

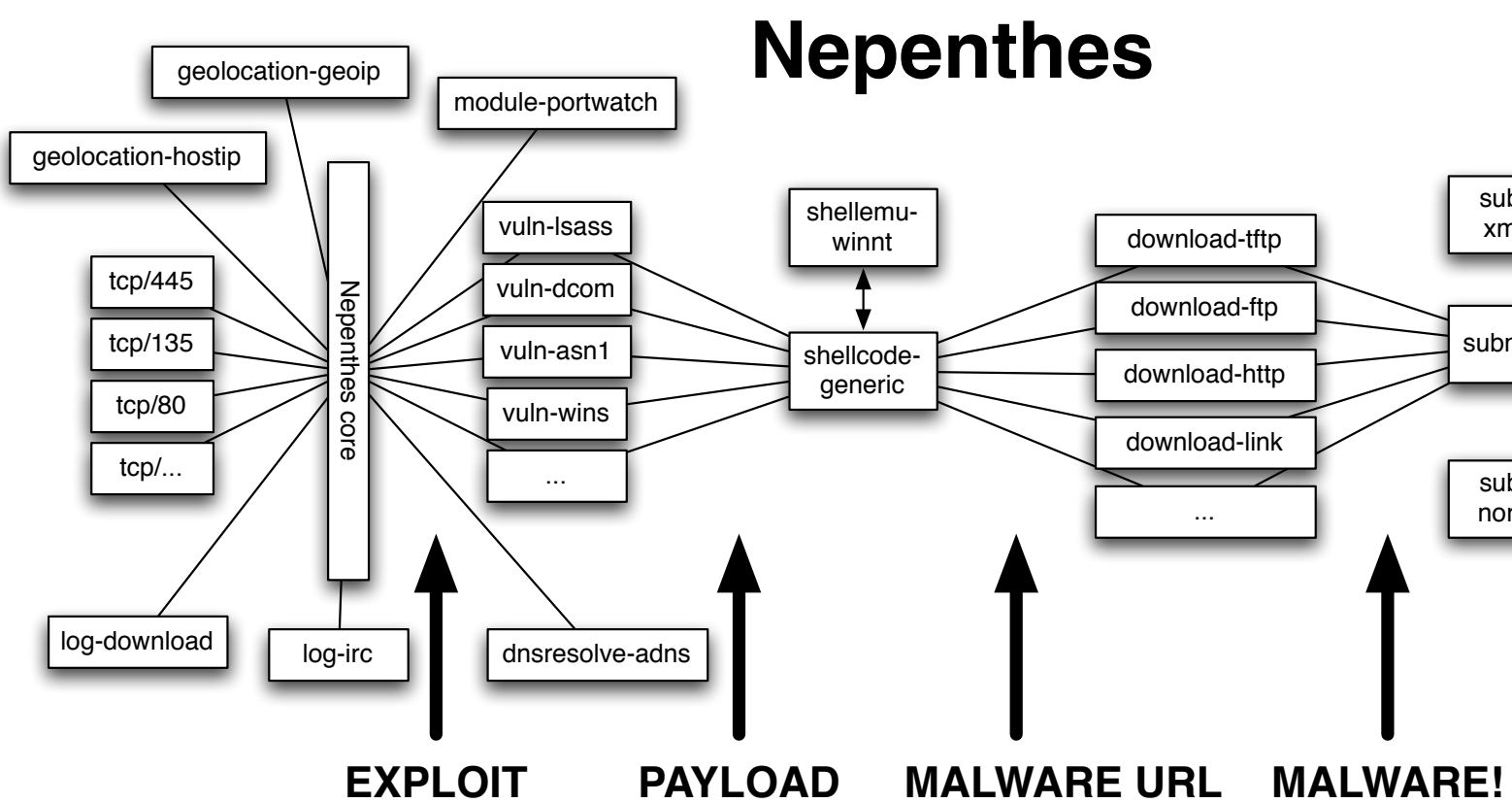
- MWCollect (sort of) is an alliance of malware researchers and software engineers
  - ...and less pretty, it is the dead parent process from which Nepenthes was forked
- Houses Nepenthes, HoneyTrap and HoneyBow
- State-of-art (scientific) research on malware
  - Reverse engineering polymorphic shellcodes
  - Call-flow graph (binary) analysis
  - Et cetera

Nepenthes.

Nepenthes architecture

- Low-interaction malware collection honeypot
- Emulates known vulnerabilities and captures the malware trying to exploit them
  - E.g. NetDDE, LSASS, DCOM, ASN1, MSSQL, UPNP, IIS vulns
- Modular arch: vuln-*, shellcode-*, download-*, submit-*
- Extensions are being developed for call-flow graphs and binary shellcode analysis
- Nepenthes is a fork() of mwcollect (way back)

Nepenthes.



HoneyTrap and HoneyBow.

HoneyTrap

- Low-interaction malware collection honeypot
- HoneyTrap handles all incoming request to unbound (!) TCP ports
- Does not simulate vulns or services, although the latter is possible through plug-ins
- Suitable for zerodays (unlike Nepenthes)

HoneyBow (future)

- High-interaction malware collection honeypot
- Fairly new: announced in Dec/2006 by China Honeynet Project (no code yet)
- Modular arch: MwWatcher, MwFetcher, MwSubmitter
- Will be interoperable with Nepenthes

Outline

Limitations.

Limitations/caveats to honeypot technology

- Complexity is the enemy of security, and honeynets are complex.
  - Bugs in emulators
  - Privilege escalation
- Known attacks: NoSEBrEaK, Phrack #62/0x07.
- Decoy/false attacks (counter-counter).
- Blackhats exchange and evade IP-ranges of known honeynets
  - Auto(re)configuration, higher volatility should help

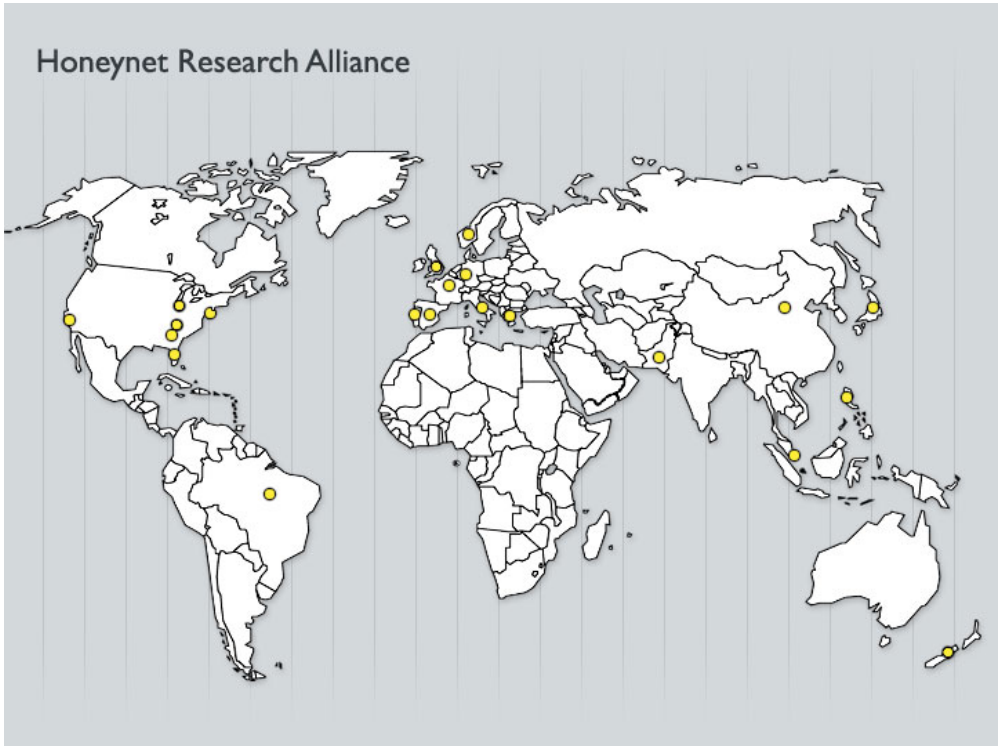
Outline

Honeynet Research Alliance.

Honeynet Research Alliance

- “The Honeynet Research Alliance is a trusted forum of other honeypot research organizations. [...] These organizations subscribe to the Alliance for the purpose of researching, developing and deploying honeypot related technologies and sharing the lessons learned."

Honeynet Research Alliance (map).



NL is not represented. Perhaps OS3/SNE should endeavor to?

Outline

Matthijs Koot
(koot@uva.nl)

Future topics.

- Honeysnap
  - CLI tool for high-level analysis of honeynet data
- Unified Data Analysis Framework (UDAF)
  - Library for data acquisition, filtering, fusion, reporting, et cetera
  - Towards visual programming
  - Let's hope it'll be interoperable with IDMEF / GOTEK
- CWSandbox (Carsten Willems)
- SCADA honeynets
  - Cisco CIAG: scadahoneynet.sf.net
  - PLC simulation; MODBUS, DNP

Summary

Topics that have been discussed

- Definitions and purpose
- Past, present, future
- Most important: HoneyD, Honeynet/HoneyWall, Sebek, Nepenthes
- Limitations

Matthijs Koot
(koot@uva.nl)

Feedback!

Question

Questions regarding this lesson?