ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Generating signatures for zero-day network attacks

Pascal Gamper

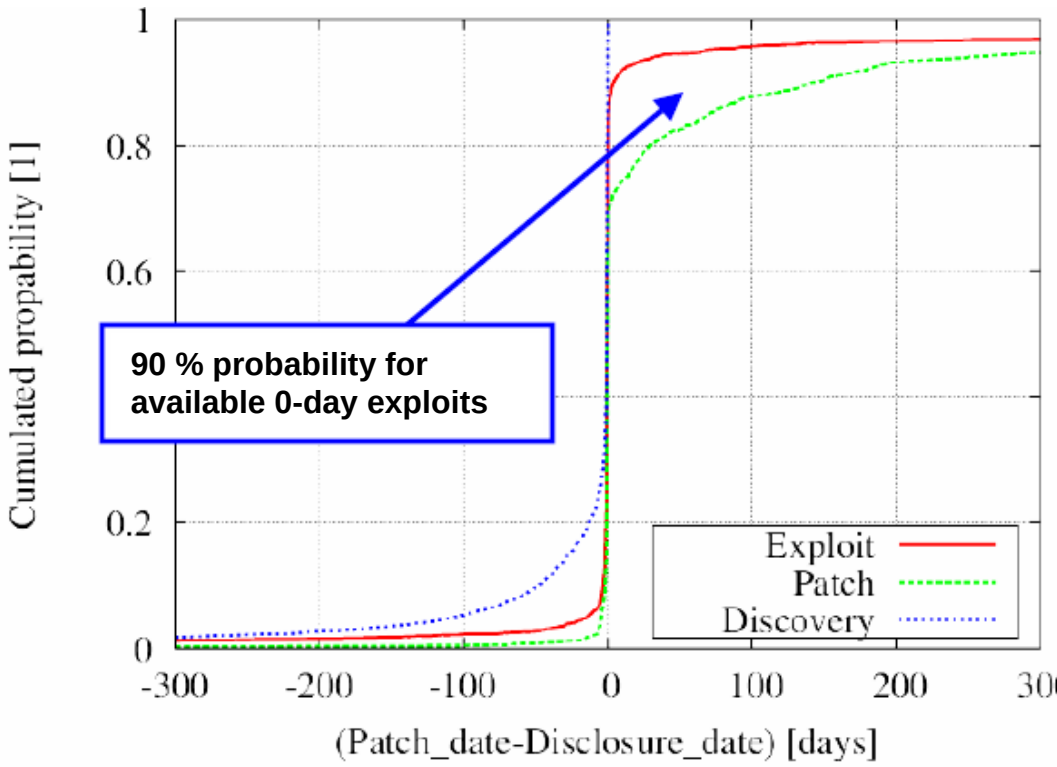
Daniela Brauckhoff

Bernhard Tellenbach

Outline

- Motivation
- Problem Statement
- Automated Signature Generation – Overview
- The NoAH approach
  - Attack Detection
  - Attack Analysis
  - Signature Generation

Motivation: The dynamics of (In)security



Y-Axis:
cumulated probability for exploit- and patch-availability dates

X-Axis:
days from disclosure-date

Data:
- 3416 exploits
- 1477 patches
- from 1996-2006

Source: “The Dynamics of (In)Security“, Stefan Frei, ETH Zurich, BlackHat 2006

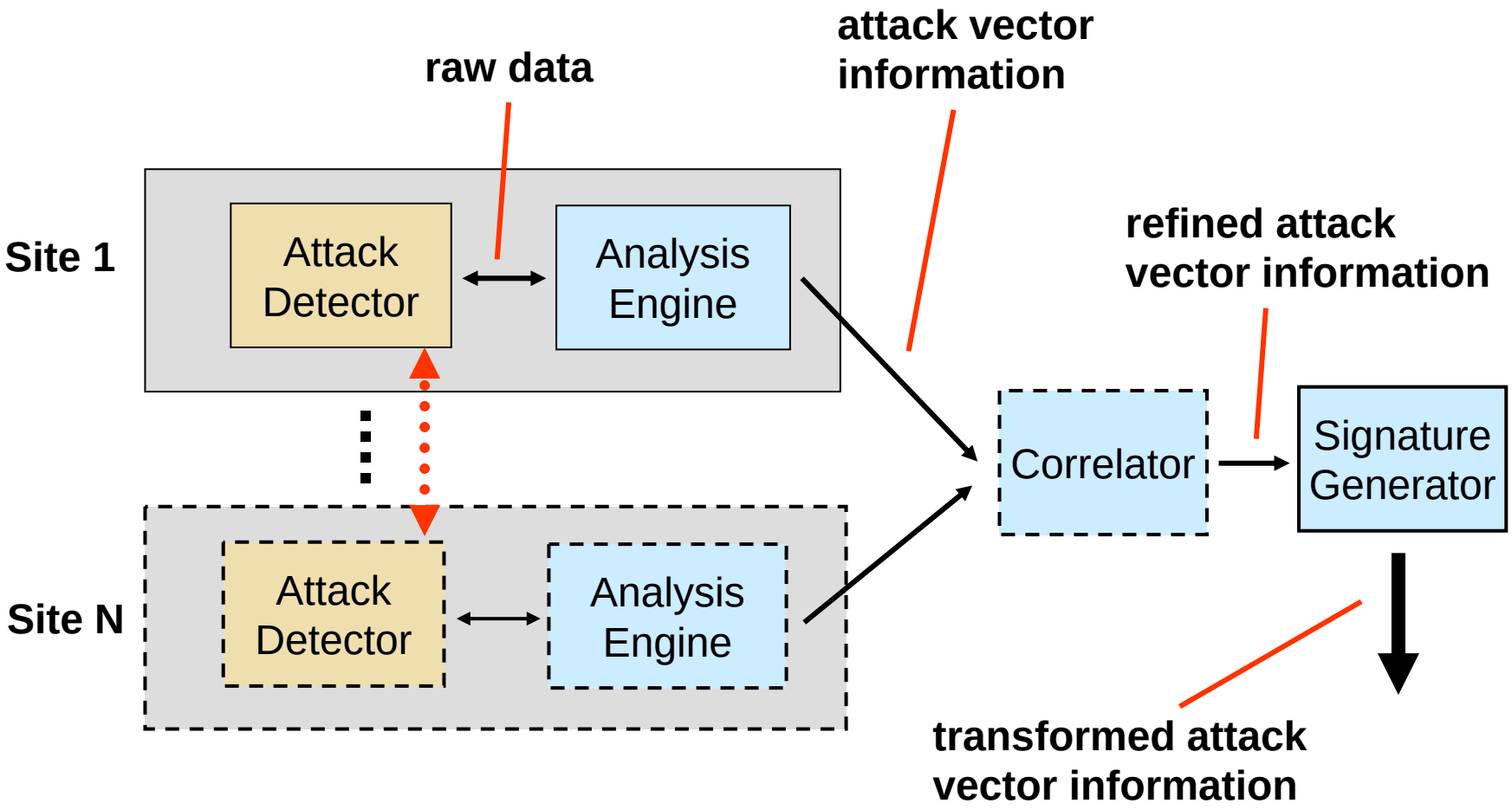
Problem Statement

- Defending against 0-day attacks: Intrusion Detection System (IDS)
  - Separate benign and malicious network traffic
  - Host- or Network-based signatures
  - Most signatures for IDS are hand-craftet by professionals
  - Zero-day exploits make manual signature generation useless
- Problem: Manual signature generation is too slow! Options?

Overview

Techniques for automated signature generation

Building Blocks of an ASG System



The NoAH Approach

EU Project NoAH (Network of Affined Honeypots)

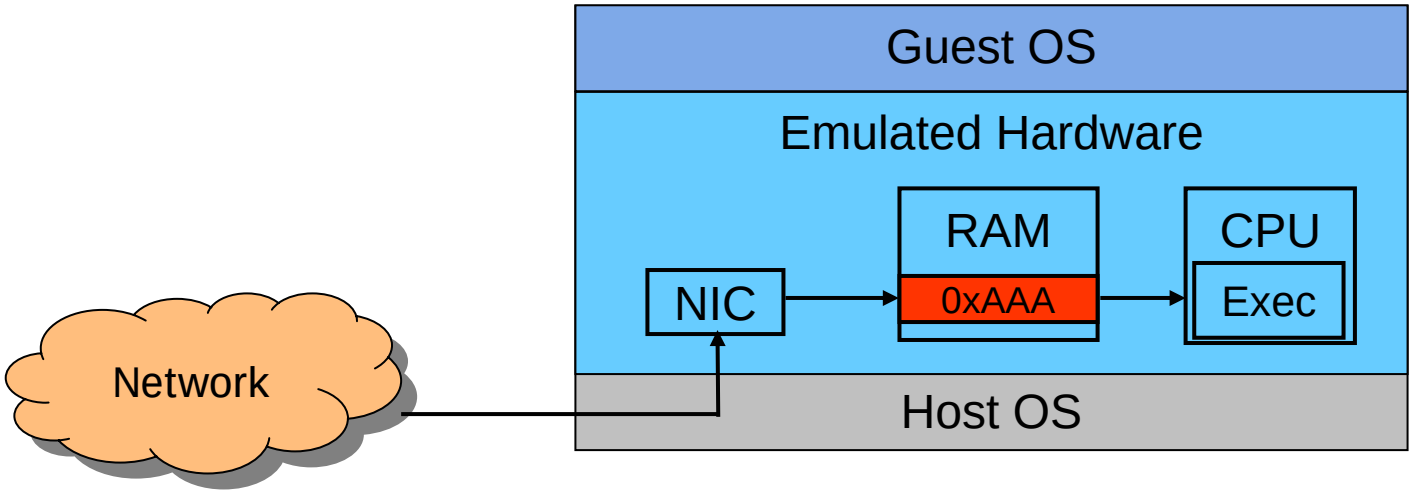
Goals

- NoAH aims at automated
  - detection of unknown attacks
  - generation of signatures to counter 0-day attacks
- Generate signatures for common IDS
- Install full-scale infrastructure across Europe
- Target audience: ISP‘s, NREN‘s, researchers

Attack Detection

NoAH Architecture: Attack Detector Argos

- Detection technique (Argos):
  - OS independent memory tainting (x86 emulator)



> Scope of NoAH: Remote attacks that do not require a human in the loop

Attack Analysis

Combining Analysis Engines

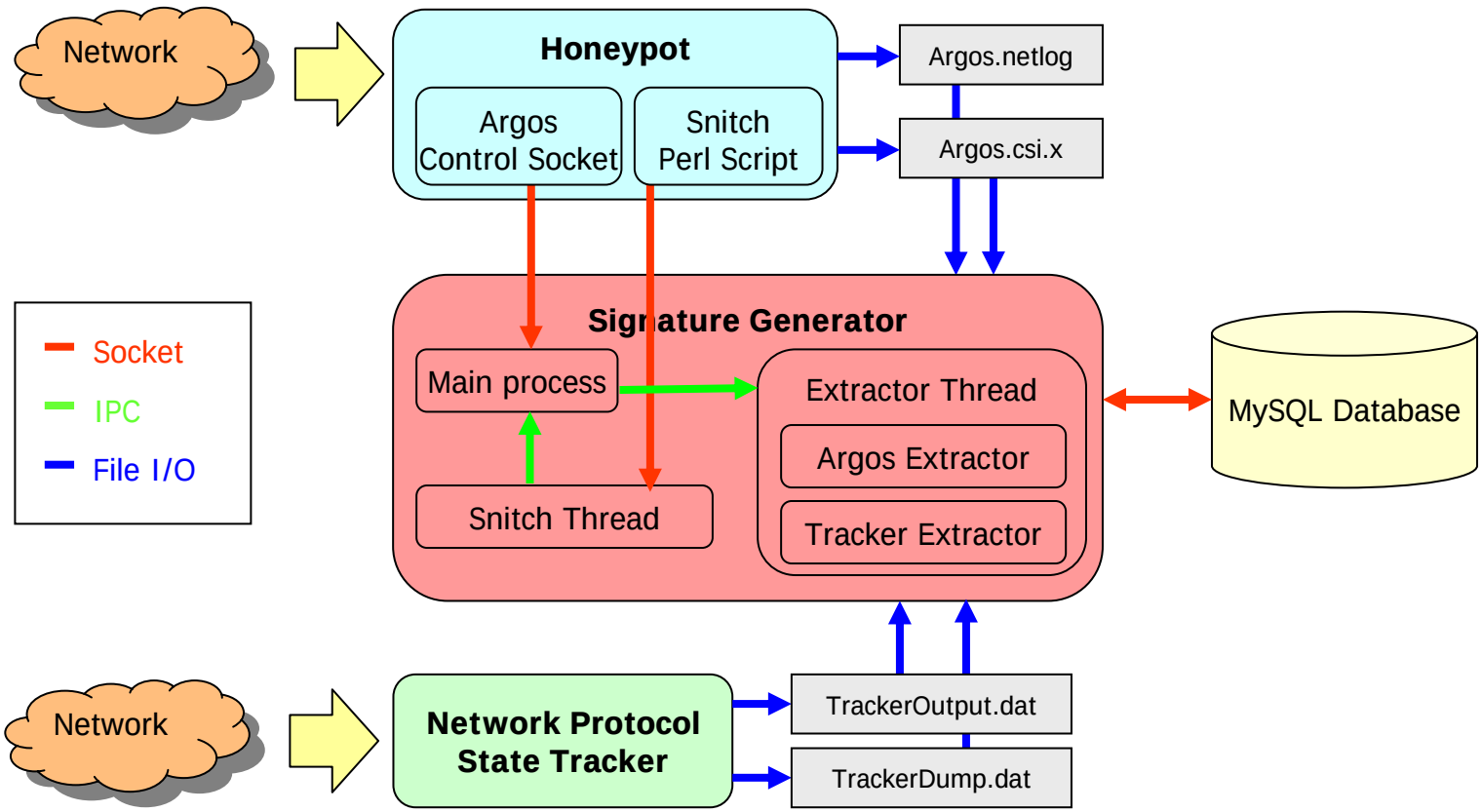
- Different analysis engines (Extractors) which
  - Analyse attack information from different sources
- Extractors can depend on each other
- Meta-signature describes entire set of available attack information
- Quality estimation of meta-signature based on
  - Which extractors succeeded
  - Value and amount of extracted information

Combining host- and network-based analysis

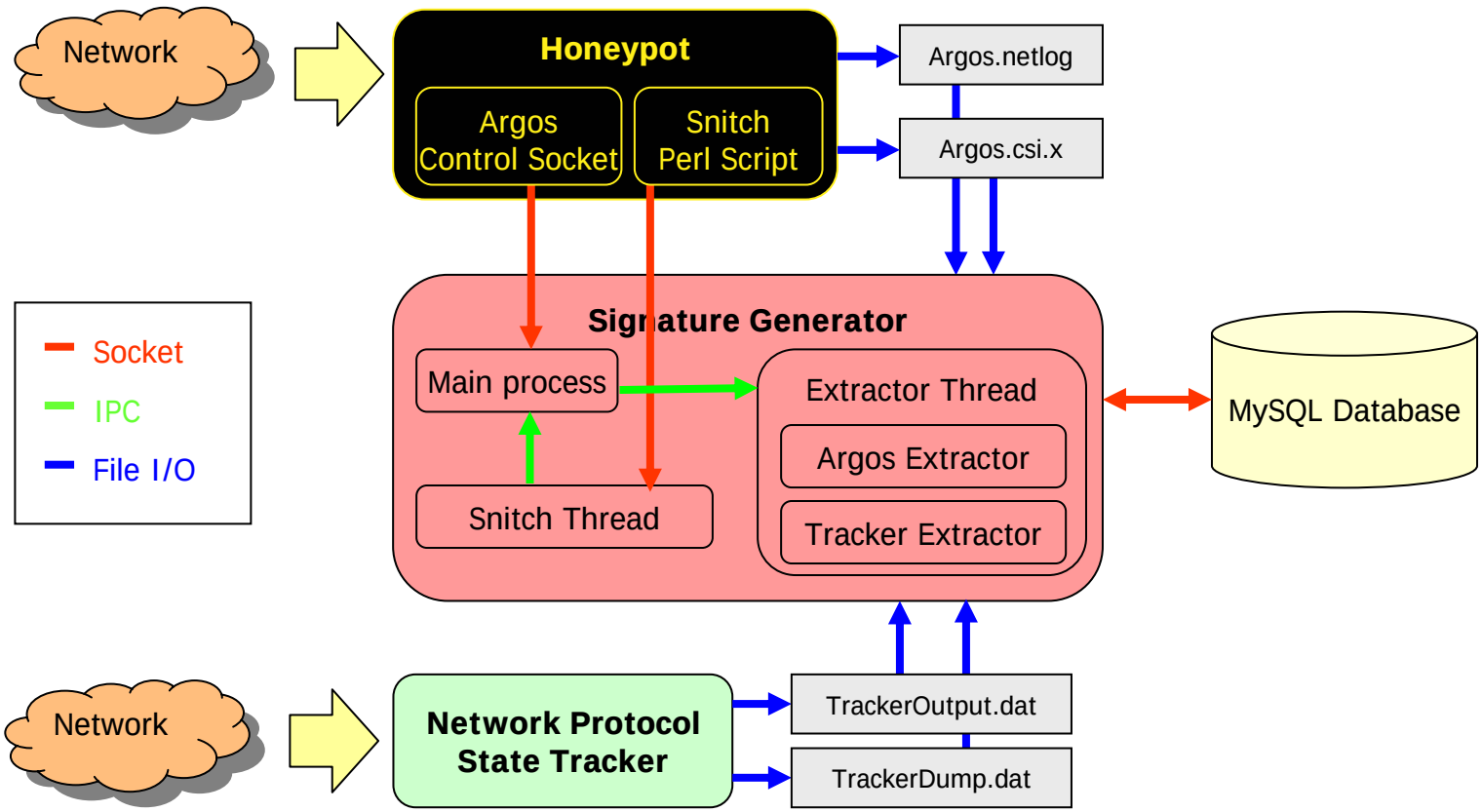
- Extractor #1: Host-based information from Argos
  - Identifies memory content relevant for the attack
  - Identifies OS and attacked process
  - Identifies network traffic bytes involved

- Extractor #2: Network-based information from Protocol State Tracker
  - Protocol field(s) containing network bytes involved
  - Communication/Protocol state history

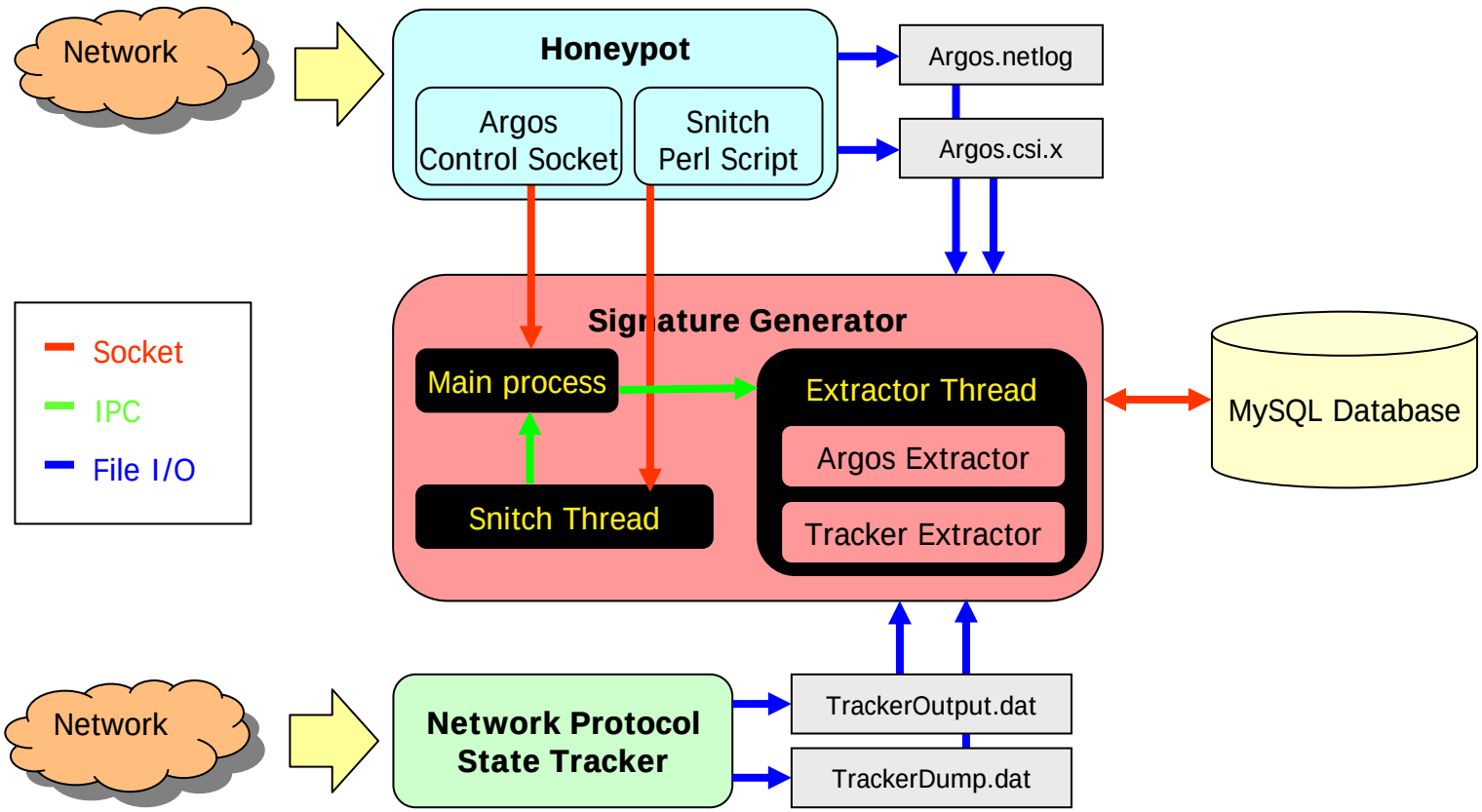
Basic Architecture of the entire ASG System



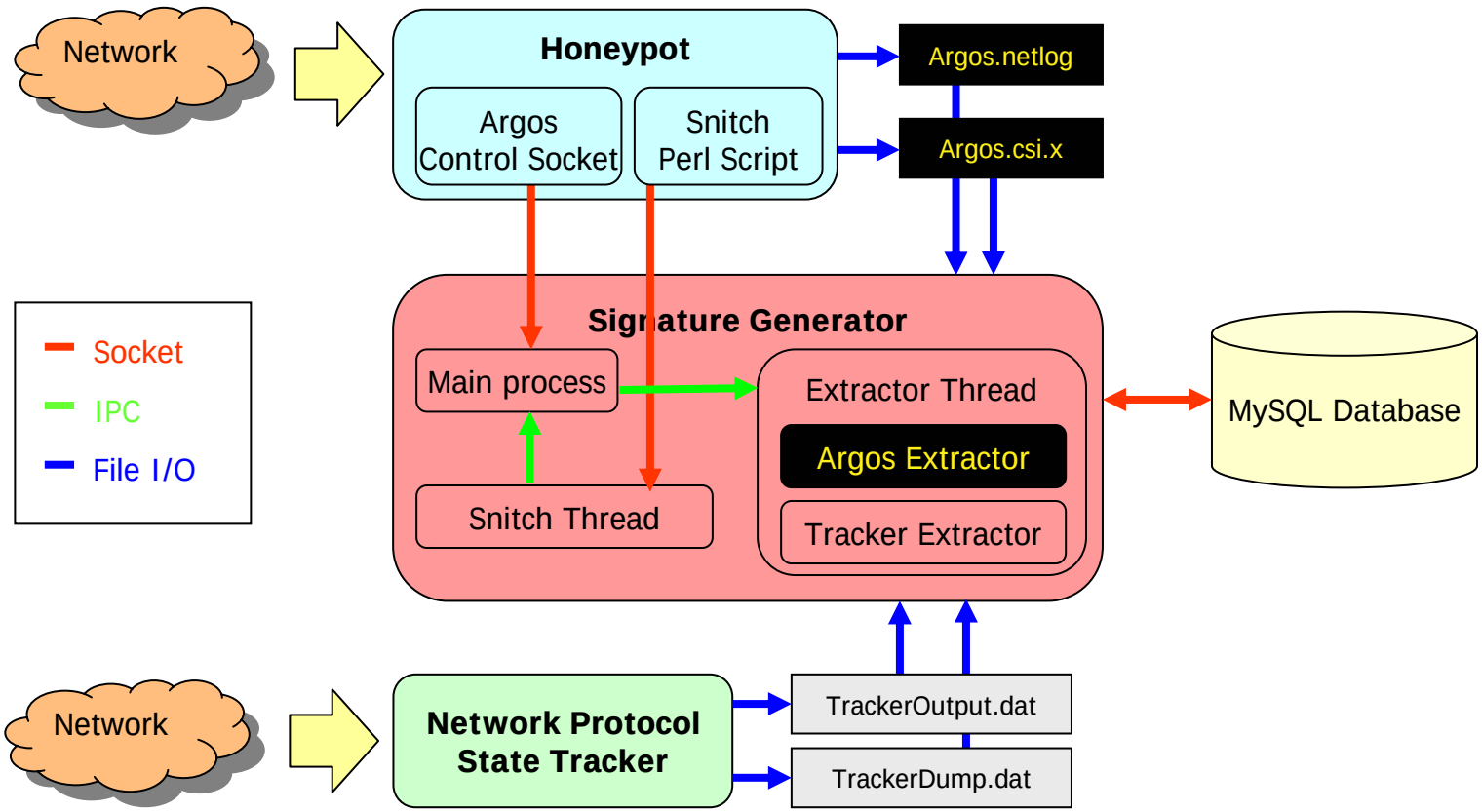
Basic Architecture of the entire ASG System



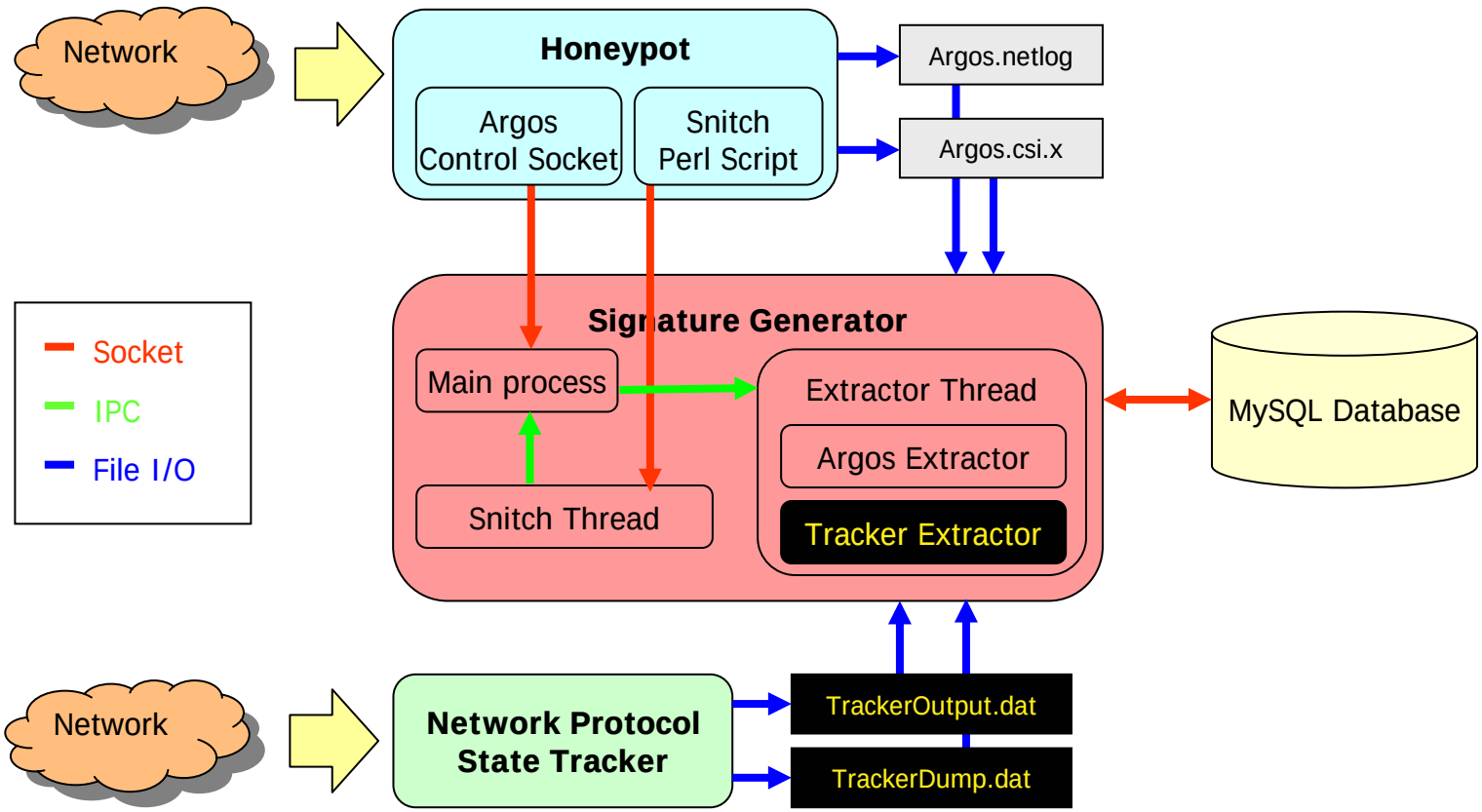
Basic Architecture of the entire ASG System



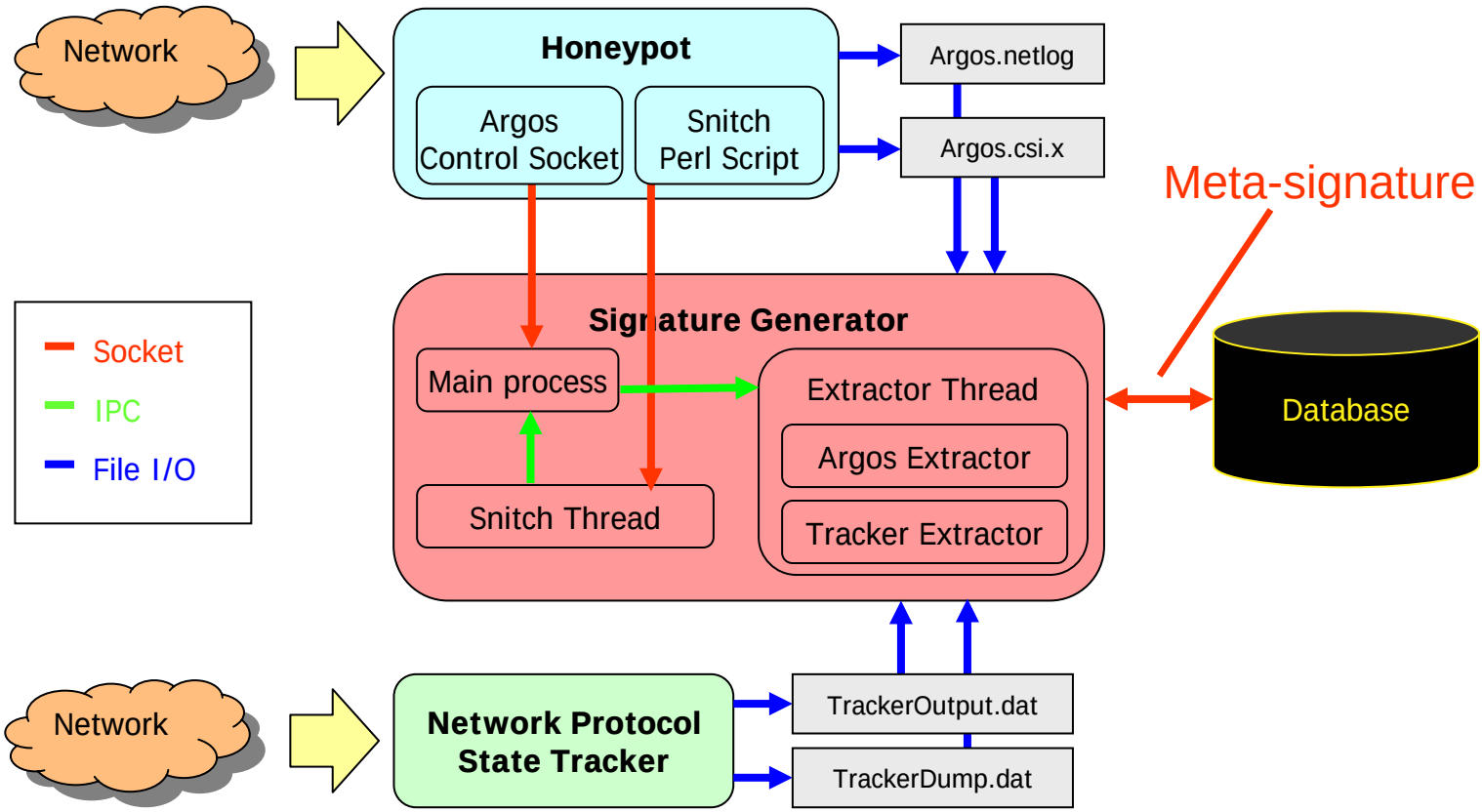
Basic Architecture of the entire ASG System



Basic Architecture of the entire ASG System



Basic Architecture of the entire ASG System



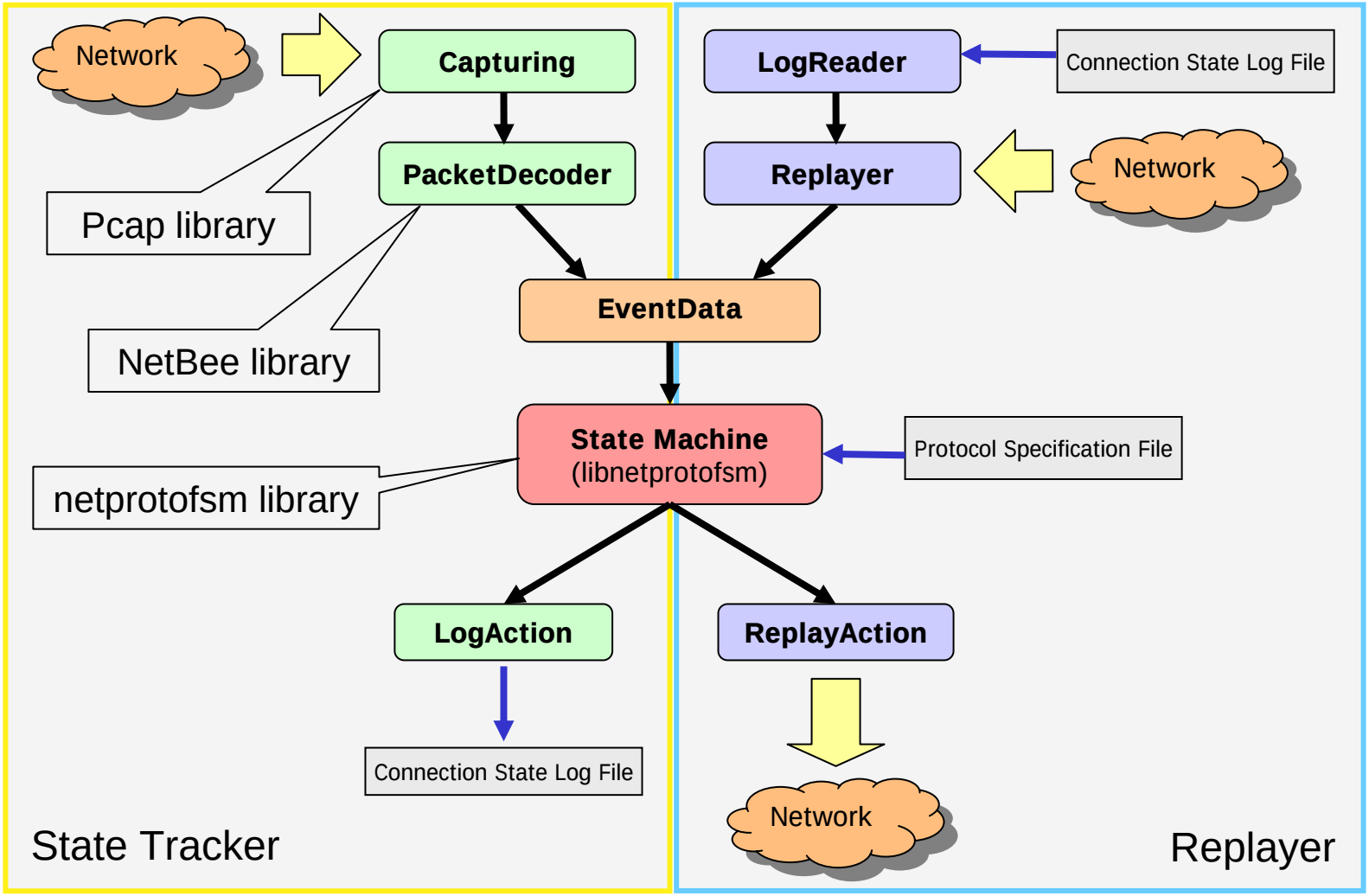
Network Protocol State Tracker

- Tracks the network connections towards one or more honeypot systems
- Logs protocol states for each packet
- User-defined packet and connection analysis possible
- Is highly configurable by relying on various libraries
- State machine configurations currently available for IP, TCP/UDP, FTP

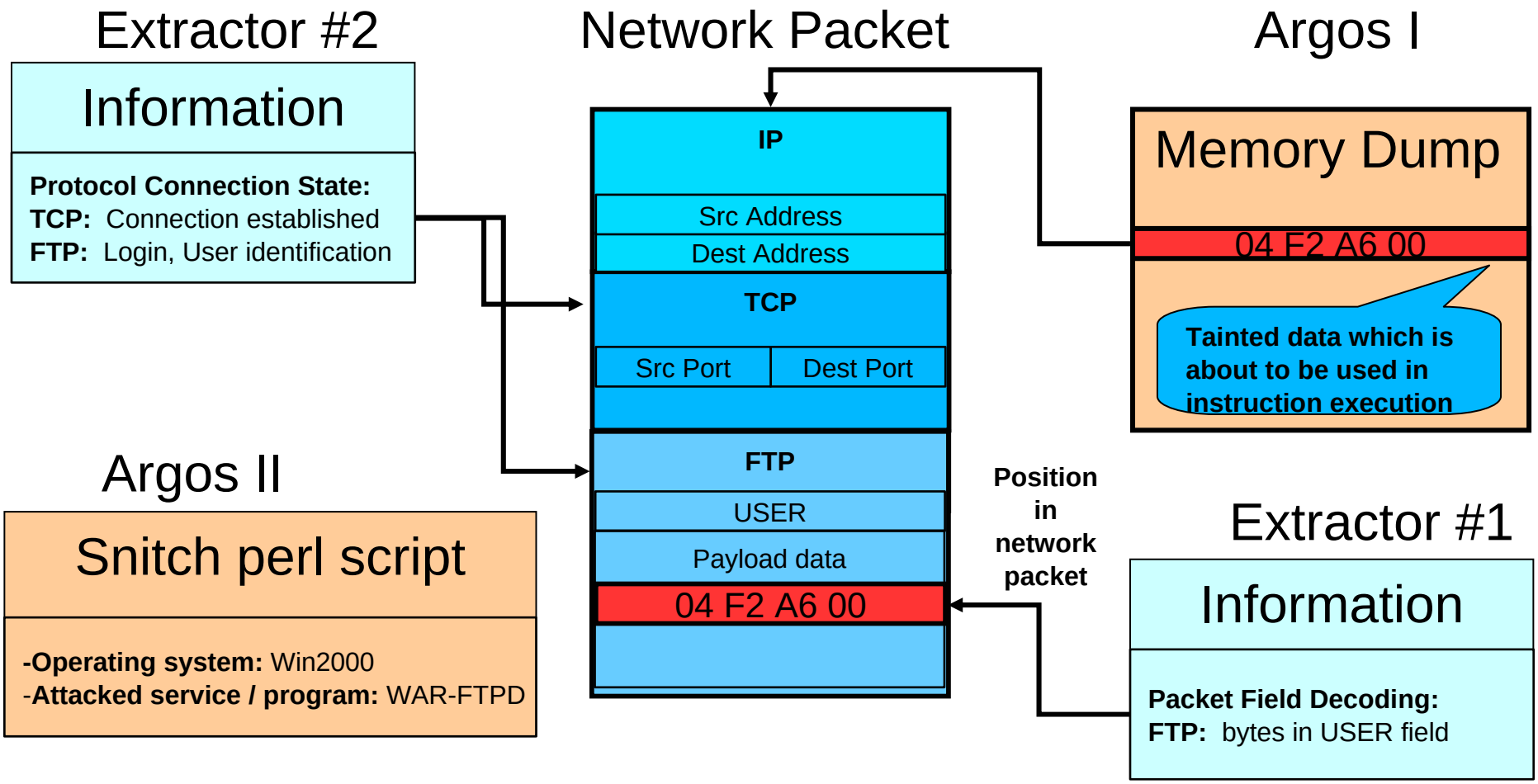
Libraries

- NetBee library
  - Developed by NetGroup at Politecnico di Torino
  - Components for different types of packet processing
  - We integrated Packet Decoding functionality into Tracker
- Netprotofsm
  - Finite state machine library for describing network protocols
  - Our approach is based on work by J. van Gurp and J. Bosch
  - Features:
    - Protocol state machines defined by XML files
    - Resource-gentle
    - Flexible timer mechanism (schedule events, define timeouts)
    - Implement custom actions

Architecture



Example: Attack information extracted



Signature Generation

Signature Generation Flow

- 1. Generate meta-signature
- 2. Determine signature quality
- 4. Save to database
- 5. Use Adapters to create specific signatures
- 6. Store, (correlate and/or distribute) adapted signatures

Snort as Signature Format

- SNORT for Proof-of-Concept
  - SNORT is open source and well-known
  - Simple signature format
- Implications
  - Only a part of extracted attack information can be used, for example
    - We cannot include information about attacked program

Generated signature (WAR-FTPD example)

```
alert tcp any any-> any 21 (msg:
“(NoAH) RET via FTP protocol, USER
command in war-ftpd.exe(win2k)”;

flow: established, from_client;
content:"USER";
content:!"|0D 0A|"; offset: 5; depth:
465;)
```

Signature Properties (WAR-FTPD example)

Connection state information

```
alert tcp any any-> any 21 (msg:
“(NoAH) RET via FTP protocol, USER
command in war-ftpd.exe(win2k)”;

flow: established, from_client;
content:"USER";
content:!"|0D 0A|"; offset: 5; depth:
465;)
```

Signature Properties (WAR-FTPD example)

State transition trigger

```
alert tcp any any-> any 21 (msg:
“(NoAH) RET via FTP protocol, USER
command in war-ftpd.exe(win2k)”;

flow: established, from_client;
content:"USER";
content:!"|0D 0A|"; offset: 5; depth:
465;)
```

Signature Properties (WAR-FTPD example)

```
alert tcp any any-> any 21 (msg:
“(NoAH) RET via FTP protocol, USER
command in war-ftpd.exe(win2k)”;

flow: established, from_client;
content:"USER";
content:!"|0D 0A|"; offset: 5; depth:
465;)
```

Vulnerable field(s)

Conclusion

- Our ASG system generates signatures with almost zero false positives
  - For remote code injection attacks
  - If full amount of attack information is extracted
- Signature describes the vulnerability of the application
- Protect server applications from buffer overflows in arbitrary protocols and fields

> Our signatures can compete with other approaches including manually created reference signatures

Questions?

Evaluation

- Prototype implementation
  - IP, TCP, UDP, FTP protocol state machines
- Examplary signature generation tests
- Protocol context aware signatures:
  - Average total generation time: 1,64 s
  - Few false positives
- LCS signatures (fallback strategy):
  - Average total generation time: 3,46 s
  - High rate of false positives depending on strings