CLASSIM

Auto-Identification/Classification of Common IP Protocols

ACCIPP

Members:
Elvan GULEN
Cagla CIG
Can HOSGOR
N. Ilker ERCIN

Supervisor:
Cagatay CALLI

Company:
Siemens

Outline

- Problem Definition
- Why ACCIPP is needed?
- Architectural Details of ACCIPP
- Current Progress

Problem Definition

- Identification of Common IP Protocols
- Many programs send/receive data over the network
- Network admins need to know what this data contains and which protocol is used

  - Determining the protocol

  - Extracting meaningful information

Determining the Protocol

- **E-Mail**

  - SMTP, POP3, IMAP
- **Internet News**

  - NNTP
- **Instant Messaging**

  - JABBER, MSN, YAHOO
- **Web**

  - HTTP

Extracting Meaningful Data

- **E-Mail**

  - Subject, Sender, Receiver, Mail Content etc.

- **Internet News**

  - Subject, Article Content, Sender, Newsgroup etc.

- **Instant Messaging**

  - Conversation log, File transfers etc.

- **Web**

  - Visited URLs, Submitted Data, Downloaded Files etc.

Current Solutions - 1

- Port Based Protocol Identification

  WIRESHARK (*Ethereal*)

  - Successful Network sniffer
  - Classifies using *Port Number*
  - If a packet received from Port:80, it says

    “This is HTTP!!”

Current Solutions - 2

- Signature Based Protocol Identification

  - Gives only exact matches, no partial matching

What WinstonSoft Offers?

- INDEPENDENT OF PORT NUMBERS
  - Pattern Recognition

    Work only with reliable data, i.e. packet content.
  - Supports Partial Matches

    " I don't know this protocol but it looks like POP3 "
  - Machine Learning

    Improves itself over time
  - Trainable

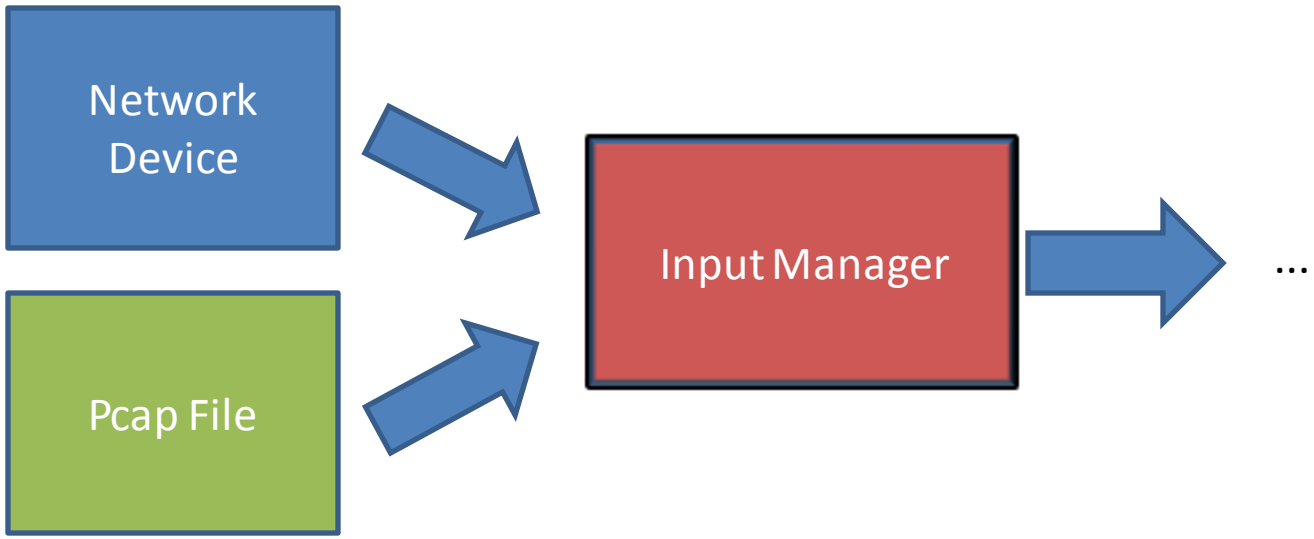
    User can assist the program in learning

Overview of ACCIPP

- Decoder Module
- Auto-Sensing Module
- Output Module

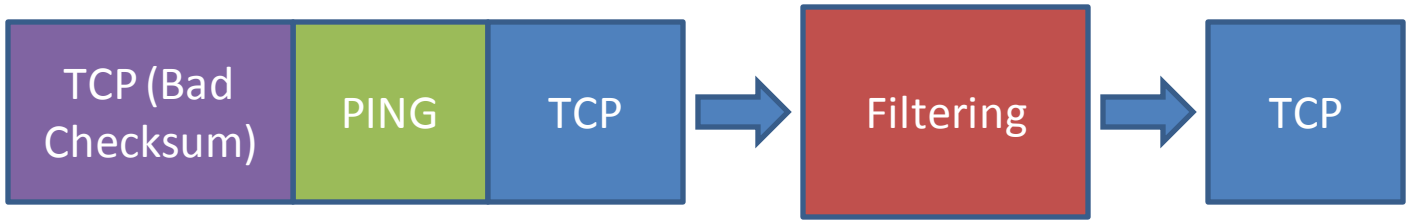
Decoder Module - 1

- Input Manager



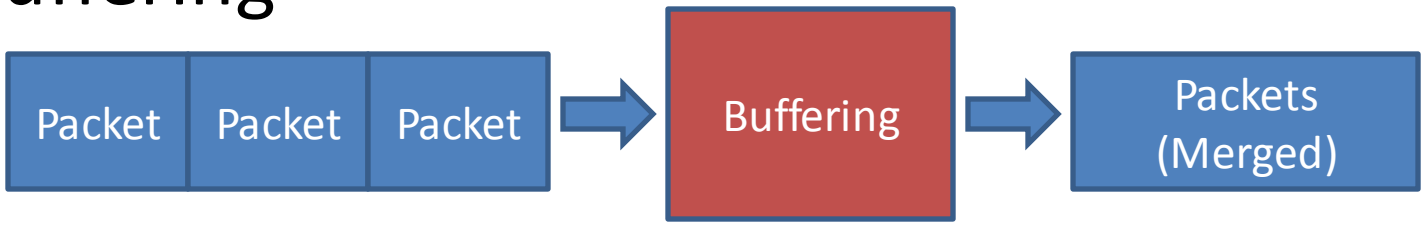
Decoder Module - 2

- Filtering



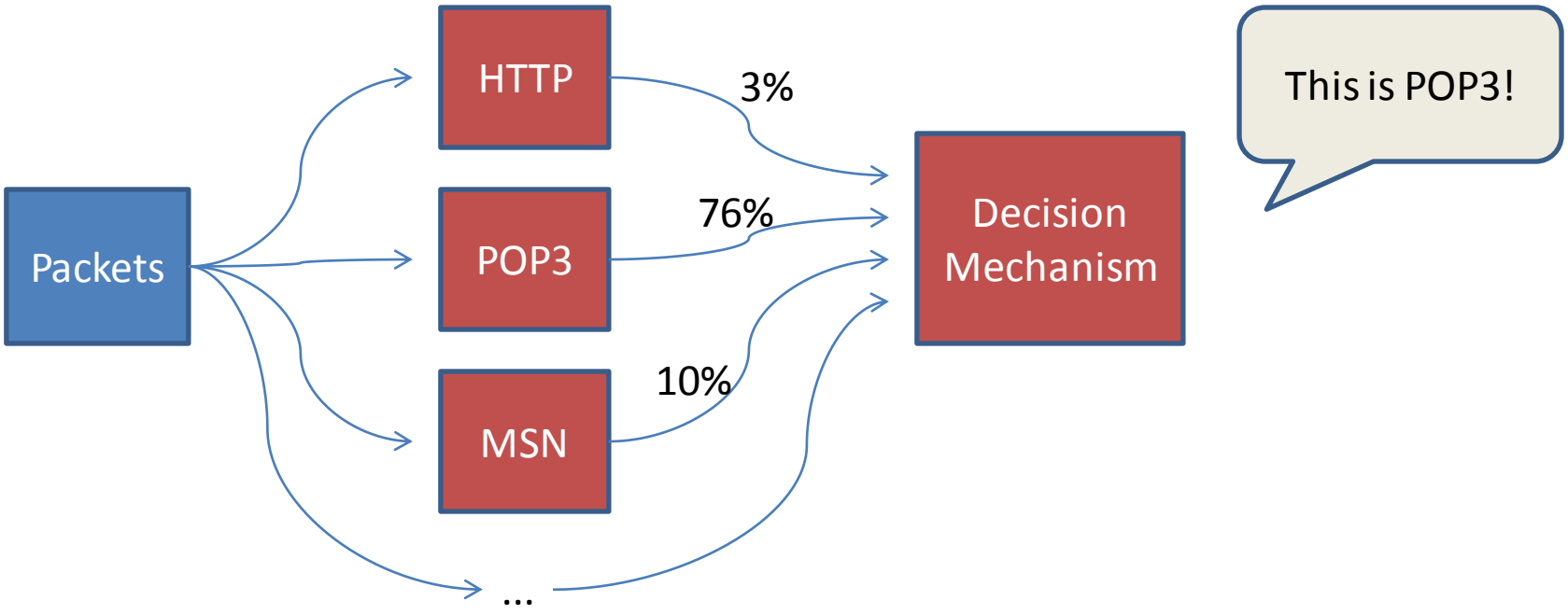
- Reordering

- Buffering



Auto-Sensing Module - 1

- Protocol Recognizers & Decision Mechanism



Auto-Sensing Module - 2

- Feedback Mechanism
  - User can correct decision mistakes
  - Protocol recognizers ***"learn"*** from mistakes

  Therefore;
  - Adapts itself to changes in protocol specs
  - More and more accurate decisions over time

Output Module - 1

- Summarizer Sub-Module
  - Extracts meaningful information.
  - Produces human-readable summary.
- User Interface Sub-Module
  - Interacts with user
  - Displays short/detailed summaries
  - Displays statistics

Output Module - 2

- Database Sub-Module
  - Stores/retrieves summaries to central datastore
  - Enables queries by user
  - Helps creating statistical info
  - Helps offline working mode

External Libraries

- Qt
  - Used in UI module.
  - Cross platform, well documented, stable
- LibPcap
  - Cross platform, de facto standard in packet capturing
- MySQL
  - Cross platform, free, easy to integrate.

Current Progress

- RFCs of protocols studied.
- GUI concept design finished.
- Decoder module almost finished.
  - Reordering part is not complete

Thank you...

Questions?