Rubicon: An Extensible Gateway Intrusion Detection System

Ian Peters

Overview

- Introduction
- Background
- Overview
- System Design – High/Low Level
- Conclusions

Introduction

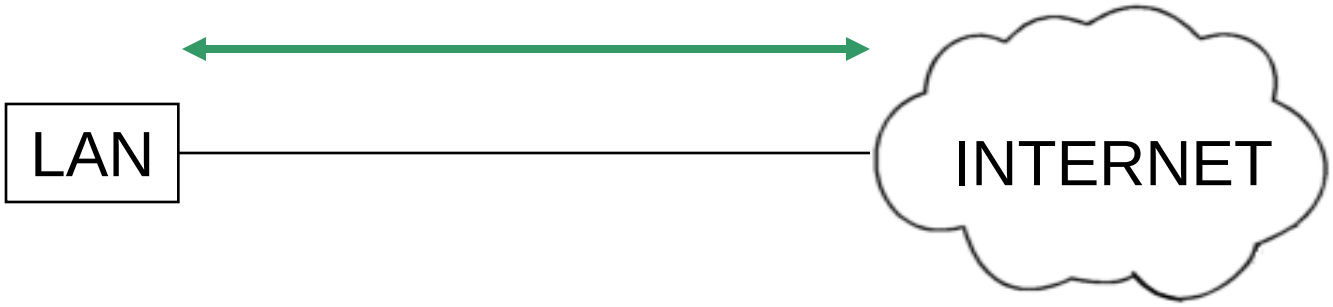
- Student at Imperial College, London
  - Now graduated (ish). Hurrah!
- Defcon 9 – Thomas J Munn (Active IDS)
- Problems:
  - Academia like algorithms not products
  - Head-crash (~3am 17 June 02)
  - Backups screwed (gz corrupted)
    - Lesson: Test your backups!
- Solution: Rewrite it!

Background

- Firewalls
- Proxies
- IDSs
- Integrating IDSs and Firewalls
- Current Technologies and Products

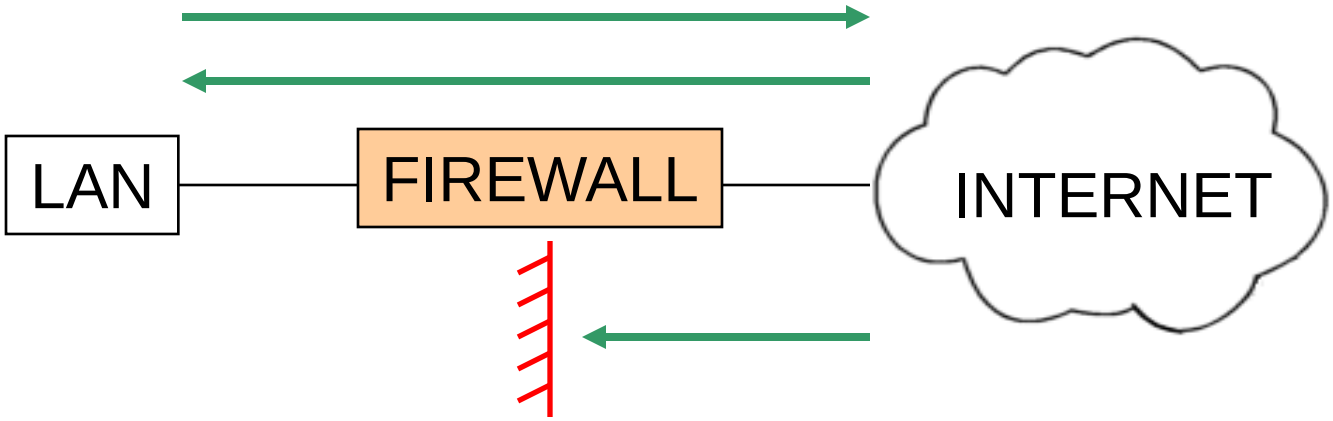
Background - Firewalls

- Naïve Design



Background - Firewalls

- Firewalled Design

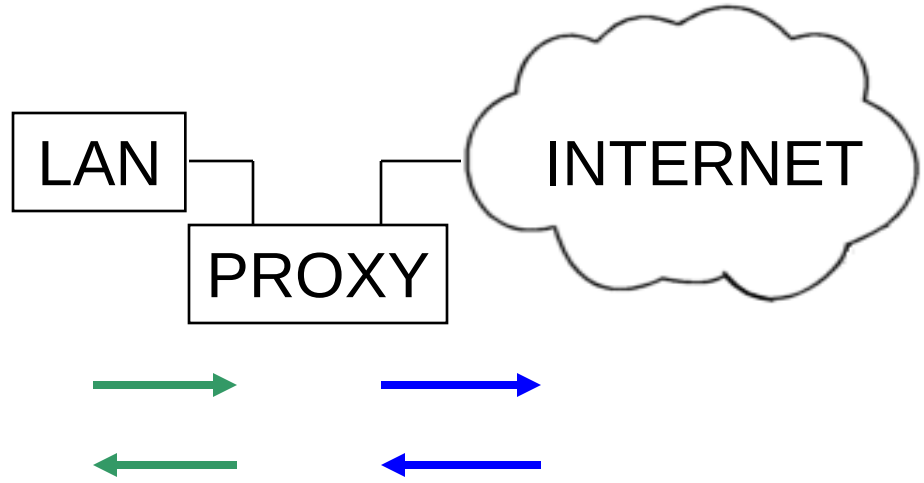


Background - Firewalls

- Firewall Types
  - Stateless
    - Makes filtering decisions based on content of each packet alone
  - Stateful
    - Tracks network state, and uses for decisions
    - E.g. Can allow ‘related’ and ‘established’ traffic
- Shortcomings
  - Cannot understand traffic data (i.e. above tcp/udp)
  - Cannot handle encrypted or some tunnelled traffic

Background - Proxies

- Application Level Filtering
  - Middle-man in transaction
  - Used for security and efficiency



Background - IDS

- No such thing as 100% secure
- Want to know about security events
- Use Intrusion Detection to gather information for human investigation
- Types – Host-based, Network-based, Hybrid
- Techniques – Pattern, Statistical, Knowledge
- False Positives – Warning about something which isn’t a threat
- False Negatives – Missing a security event

Background - IDS

- What IDSs can’t do.
  - Stop an attack.
  - Detect a wholly new attack (although newer approaches and techniques are getting better at this, e.g. Heuristics, Statistical, Anomaly)
  - Have no false positives or negatives

Background – Integration

- Allow firewall-style filtering off IDS results
- Cut the response time between Detection and Reaction (by removing the human element)
- Allow complex functionality e.g. dynamic rerouting of attackers into honeynets

Background – Integration

- Techniques: -
  - Forge control packets (e.g. TCP Reset)
  - React based on log file outputs
  - Wrap firewall outputs around IDS code
  - Insert IDS facilities into Firewall
  - Fully bespoke, integrated

Background – Integration

- Current State of the Art:-
  - Minimal research in Academia
  - Hogwash (wraps Snort IDS)
  - Several commercial offerings include some kind of response (BlackICE, Checkpoint)
  - Symantec and Realsecure have Enterprise IDS which can run commands, e.g. to modify firewall rules
  - Recently, GIDS functionality has been added by several vendors

Background – Integration

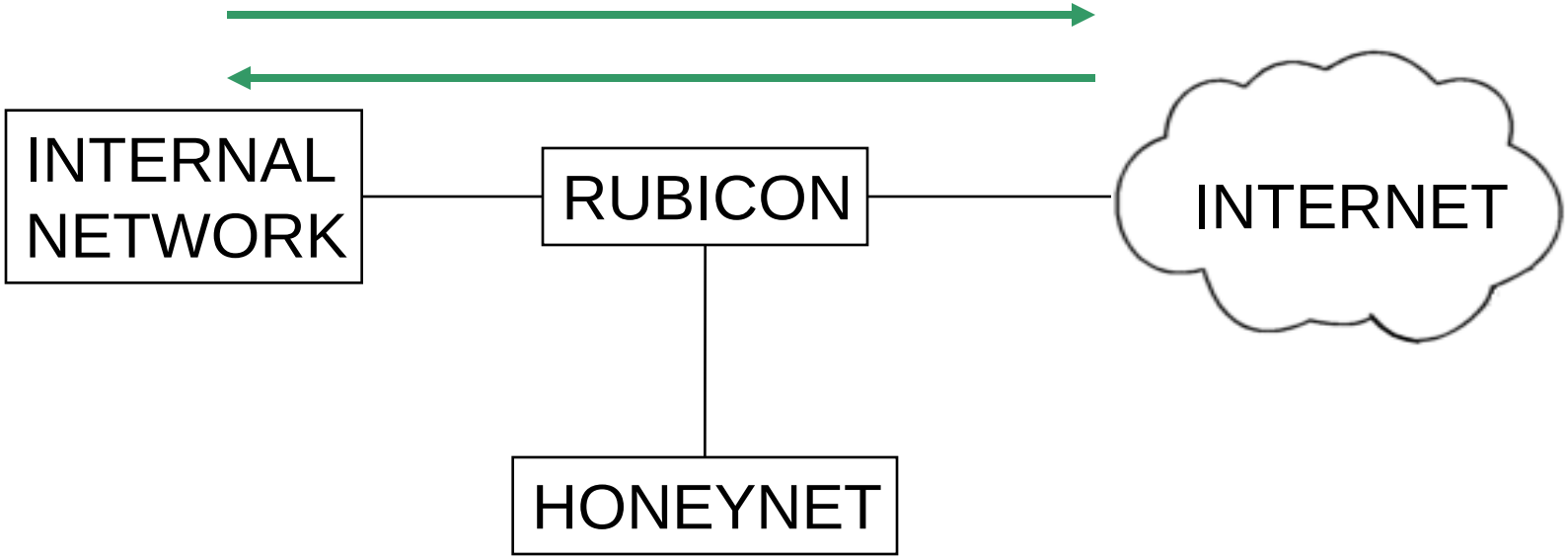
- Problems, Shortcomings & Limitations
  - False Positives
  - Denial of Service
  - Processing power and Bandwidth

Overview - Aims

- Integrate IDS and Firewall
  - Decrease Response delay
  - Allow inline packet filtering
- Modular design
  - Highly Extensible
  - Ease future development

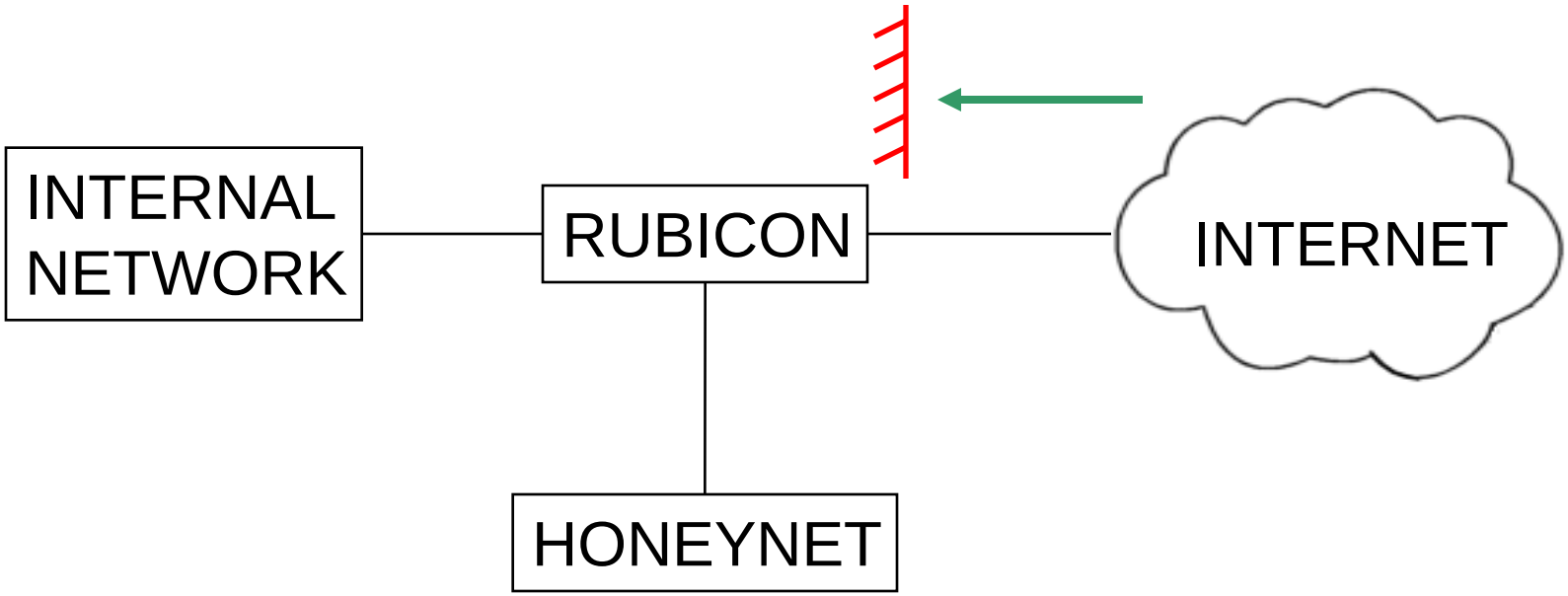
Overview – Example

- Example Situation



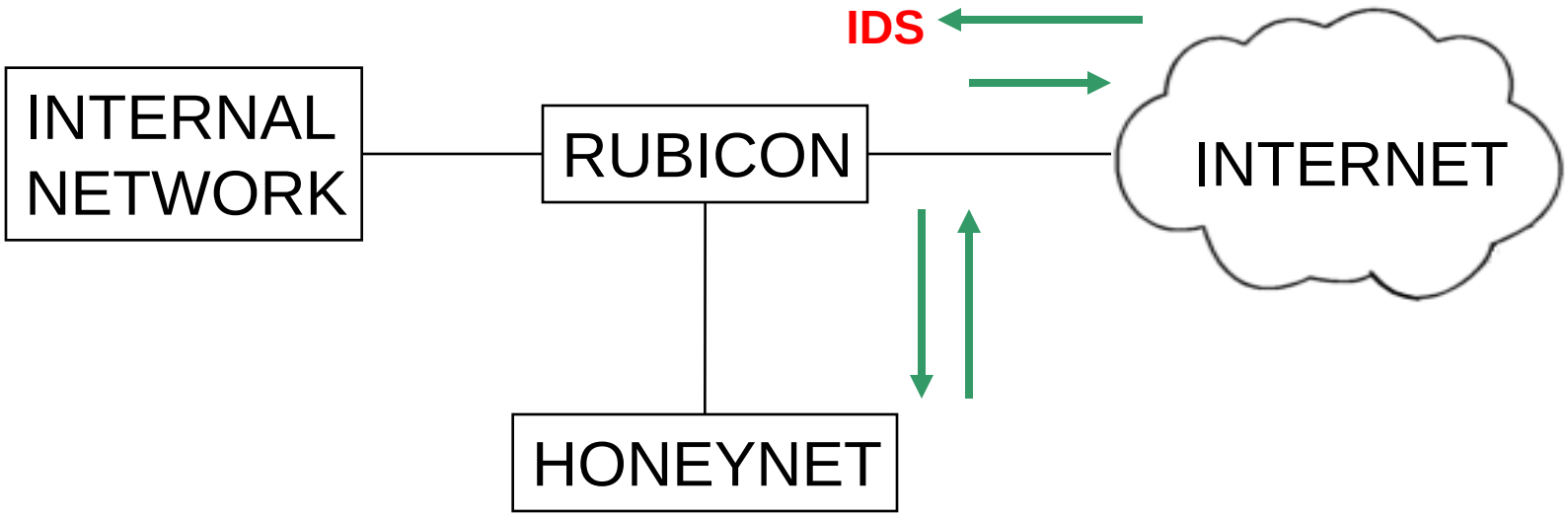
Overview – Example

- Example Situation



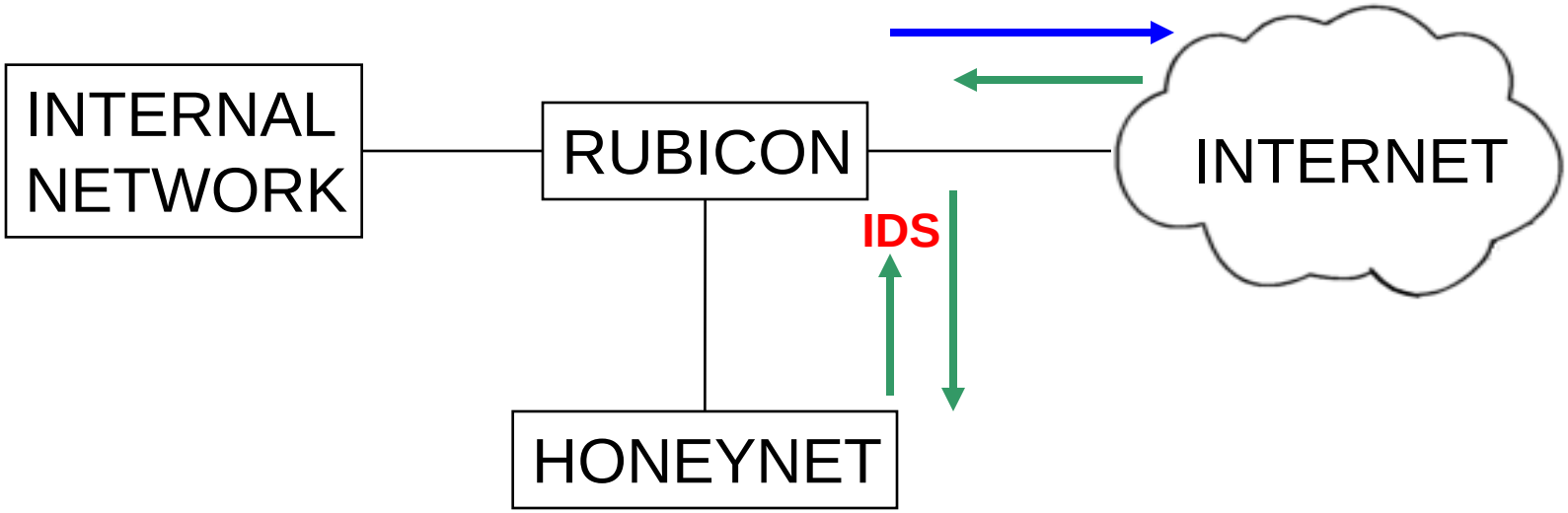
Overview – Example

- Example Situation



Overview – Example

- Example Situation

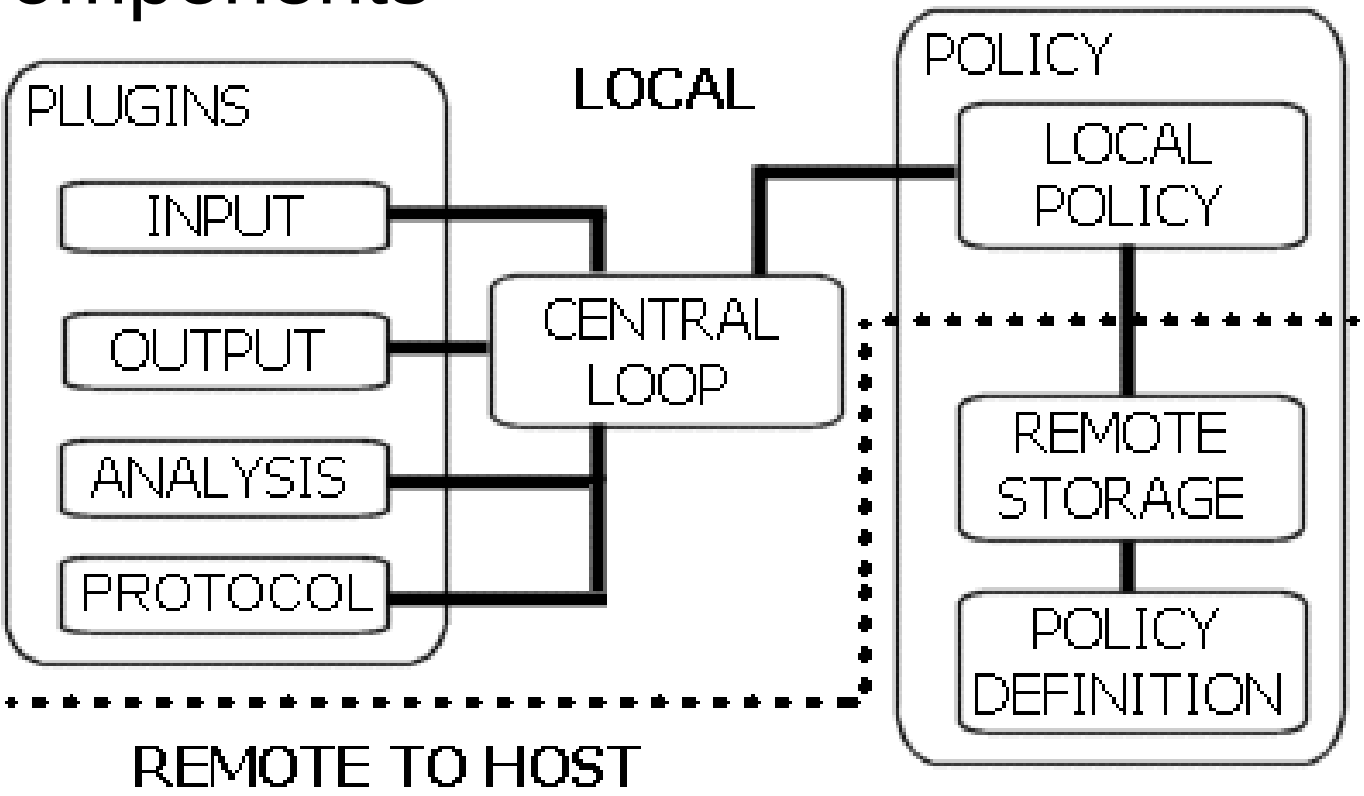


Overview – Design

- Requirements
  - Provide basic IDS and firewall functionality
  - Allow filtering based on IDS-style analysis
  - Must itself be secure
  - Must be stable
  - Must have reasonable performance

Design – High Level (HL)

- Components



Design (HL) – Core

- Loops, querying the policy and calling the required plugins
- Appends data to a PACKET structure, which wraps the packet data.
- Appended data includes PROTOCOLs
- Each protocol is uniquely named.

Design (HL) – PACKET

- Contains packet.
- Points to linked list of PROTOCOLs.
- Current pointers to packet payload etc

Design (HL) – PROTOCOL

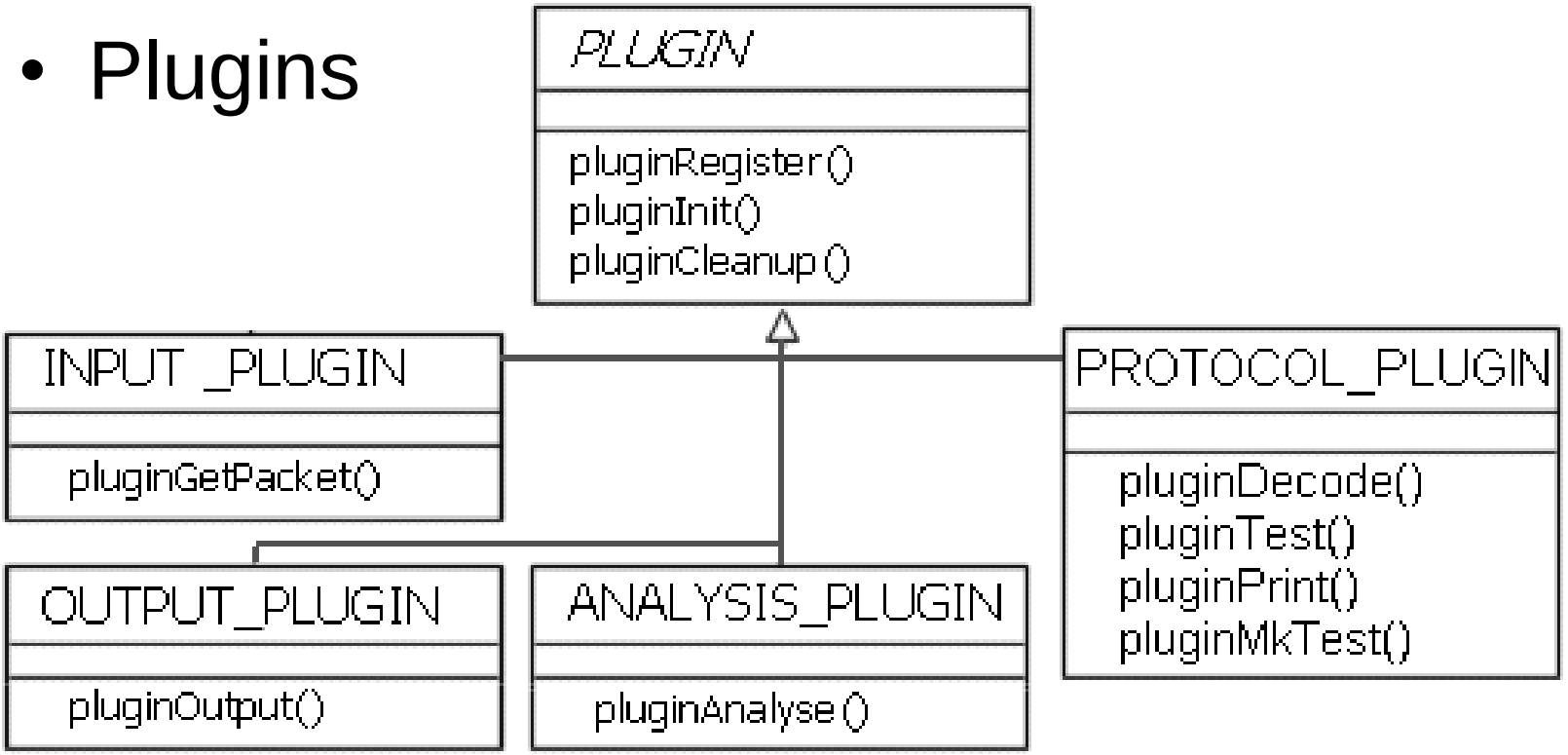
- Decoded Protocol Header (and Footer)
- Normally just a pointer to relevant part of packet
- May point to other Protocol data instead
  - E.g. decrypted traffic from SSL payload

Digression - SSL

- SSL – Encrypt traffic
- Need a common key
  - Key Exchange
    - RSA – Client encrypt Ks with Servers public
    - Diffie-Hellman – Uses combination of powers
  - Only if RSA, and IDS has servers Private key
    - Use ssldump & openssl to decode
  - Issues: Security and Processing Power

Design (HL) - Interfaces

- Plugins



Design (HL) - Interfaces

- Policy
  - Internal
    - Mutex'ed shared pointer to compiled representation
    - HandlePacket() function for handling received packets.
  - External
    - TCP Control Channel
    - LDAP Distributed Storage

Design (HL) - Policy

- High Level – Unimplemented
- Intermediate Level – XML version of low-level representation. 1:1 mapping. Stored on LDAP
- Low Level – Linked list structure

Design (LL) – Policy (Intermediate)

- Useful for compatibility
  - Byte Order, character encoding, variable size
- Human readable/write-able
- Want fast translation on Rubicon host
  - 1:1 mapping between low level structure and XML.

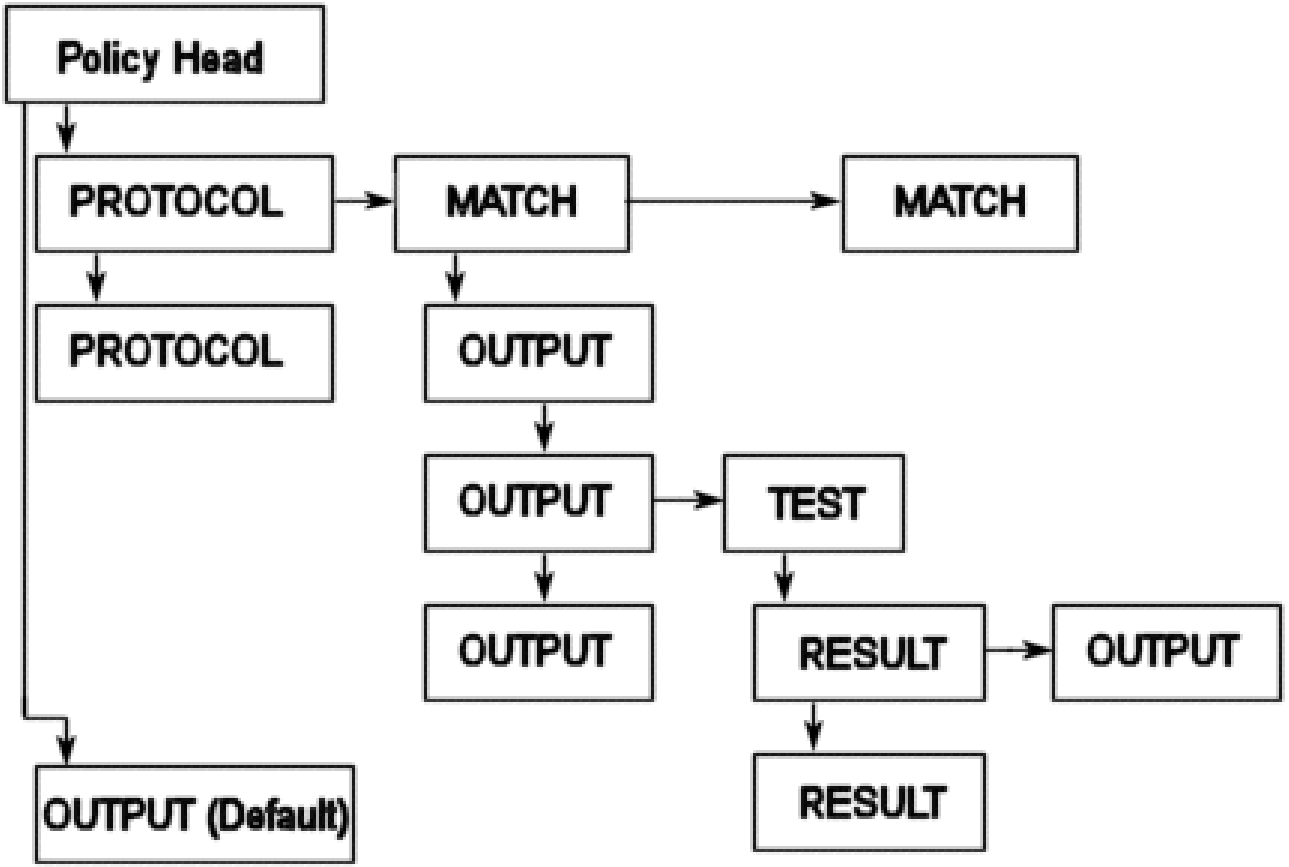
Design (LL) – Policy (Intermediate)

```
<protocol proto="ip" src="192.168.0.1">
  <analysis name="IDS" args="all">
      <result match="warning">
            <output name="FILE">WARNING!!
            </output>
      </result>
      <result match="">
            <output name="NET-FWD">
            </output>
      </result>
  </analysis>
</protocol>
```

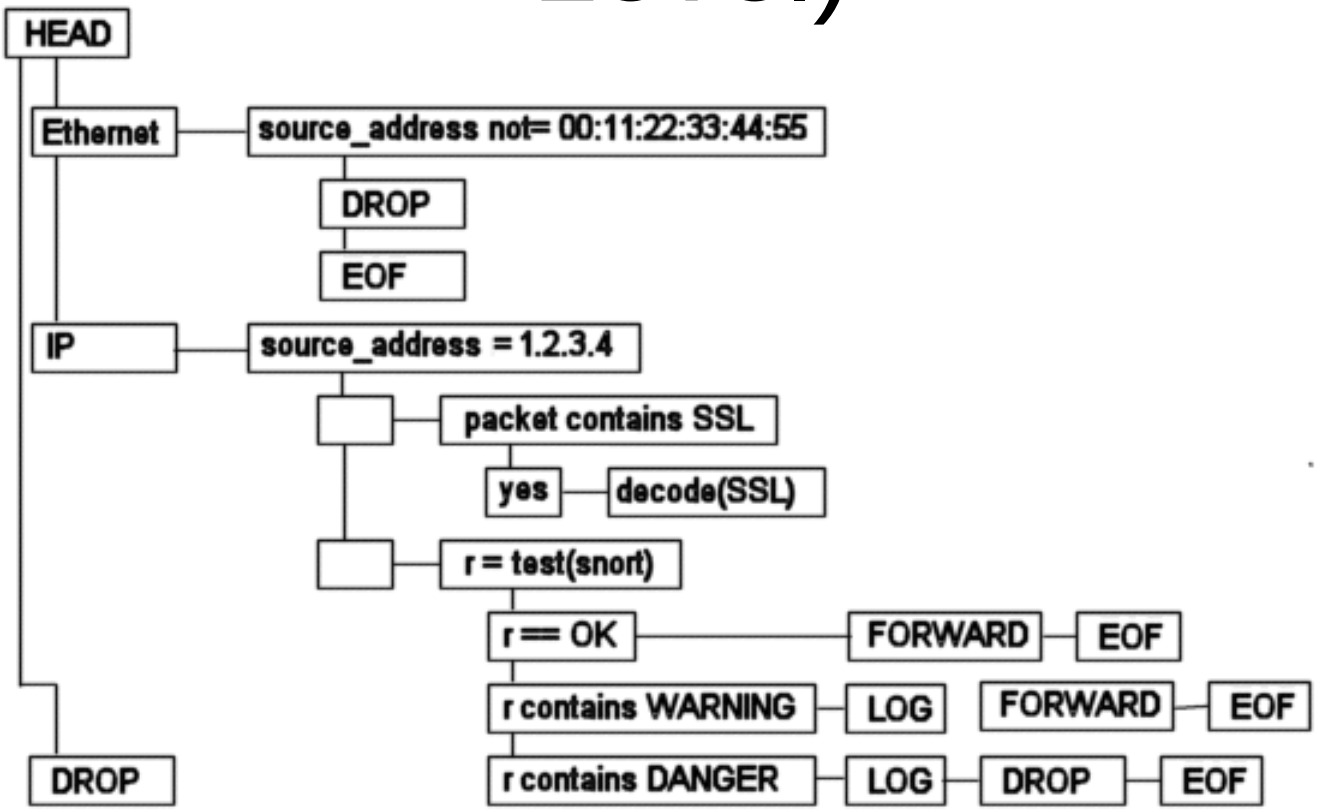
Design (LL) – Policy(Low Level)

- Linked list for fast recursion/descent
- Essentially a very large tree
- Nodes represent tests and results, leaves are outputs
- Leaves specify whether to stop processing when reached, or to continue

Design (LL) – Policy(Low Level)



Design (LL) – Policy(Low Level)



Design (LL) - Plugins

- Input (Polled)
  - Libpcap
    - Raw network device access (timeout=0,count=1)
  - Libipq
    - Receive packets for handling from netfilter/iptables
    - Both Input and Output plugin
    - Problem: Packet data copied = slow

Design(LL) - Plugins

- Protocol
  - Ethernet, IP, TCP, UDP, ICMP
    - Provides printing and test making and special tests
      - e.g. valid checksum?
- Analysis
  - Intrusion Detection (signature/pattern-matching)
    - Snort algorithms (Boyer-Moore) (match.c)
    - Initialised with tuples of (set-name, rule)
    - Call for specific set-name
      - e.g. perform IIS rules on the IIS machine

Design(LL) - Plugins

- Analysis
  - Counter
    - Allow policy to operate and test long's
    - Operators: = + - / * ^
    - Tests: == != < > <= >=
  - Defragmentation (IP and Ethernet)
    - Heavily based on snort. defrag2 plugin
    - Returns whether is fragged or not
    - If fragged, reassemble, when done place on Input

Design (LL) - Plugins

- Output (Network)
  - Provided via libnet (the Mike Schiffman one)
  - Core network (Drop, accept, reject)
    - Use human device names, and map to /dev/???
  - Net mangling
    - Modification of packet contents
    - Any payload or testable header field
    - Cannot resize due to sequence numbering

Design (LL) - Plugins

- Output (Logging)
  - Log (File logging)
    - Use printf style args to call plugins to print any testable field
  - Syslog (Unix system logger)
    - Same as log, except you can set syslog priority etc
  - IDXP (Intrusion Detection Transfer protocol)
    - Draft RFC based on BEEP(Blocks Extensible Exchange Proto)
    - Transmits IDMEF data (which is XML)
    - May even get it working one day!

DEMO TIME!!!!

(Hopefully)

Conclusions

- Benchmarks
- Problems and Limitations
- Successes
- Further Work

Conclusions - Benchmarks

- None available yet due to code rewrite
- Original version:
  - Stress Testing
    - 0% Drop @ 8MBps (dumping all to file in text)
  - IDS comparison
    - 45MB file parsed and IDS'd (SANS 2000)
    - Rubicon 2 minutes 45 seconds (approx)
    - Snort 2 minutes (approx)
    - Output was identical

Conclusions – Problems and Limitations

- Some problems accessing tunnelled traffic
- Some bugs
- Needs lots more work

- Need good uptake to be useful

BUT

Conclusions - Successes

- Modular
- Performs packet filtering, and more
- Unique functionality (mangle)
  - Hogwash apparently now does this
- Relatively cross-platform (in theory at least)
- Easy to develop plugins for

Conclusions - Further Work

- General debugging, testing, and expansion
  - Statistical IDS
  - Load Balancing
  - IntServ QoS (e.g. RSVP routing)

Conclusions - IDS

- Reduce false positives
- Pickup new attacks
- Distribute analysis, e.g. firewall NICs
- Better correlation between HIDS/NIDS
- Better standardisation
- More HIDS due to encryption
- Exploits for NIDS/HIDS

Questions ?

- Rubicon Homepage – sourceforge.net/projects/Rubicon/
- Snort - www.snort.org
- Hogwash - hogwash.sourceforge.net
- IDMEF/IDXP-www.ietf.org/ids.by.wg/idwg.html
- libidxp - idxp.codefactory.se
- libpcap - www.tcpdump.org
- iptables/libipq - www.netfilter.org
- libnet - www.packetfactory.net/libnet/
- RFCs - rfc.net