

N.E.R.D.

Sander Degen

TNO Information & Communication Technology



DESEREC

*Dependability and Security by Enhanced
Reconfigurability*



-Agenda

Introduction to NERD

- n History
- n Screenshots

Background info

- n Syslog
- n Netflow
- n Clusters & Filters

Demonstration

- n Demo of the webinterfaces of both versions
- n Explanation of the options

Technical description NERD system

- n Going in-depth after the demo, discussing technical details
- n Describing TODO list

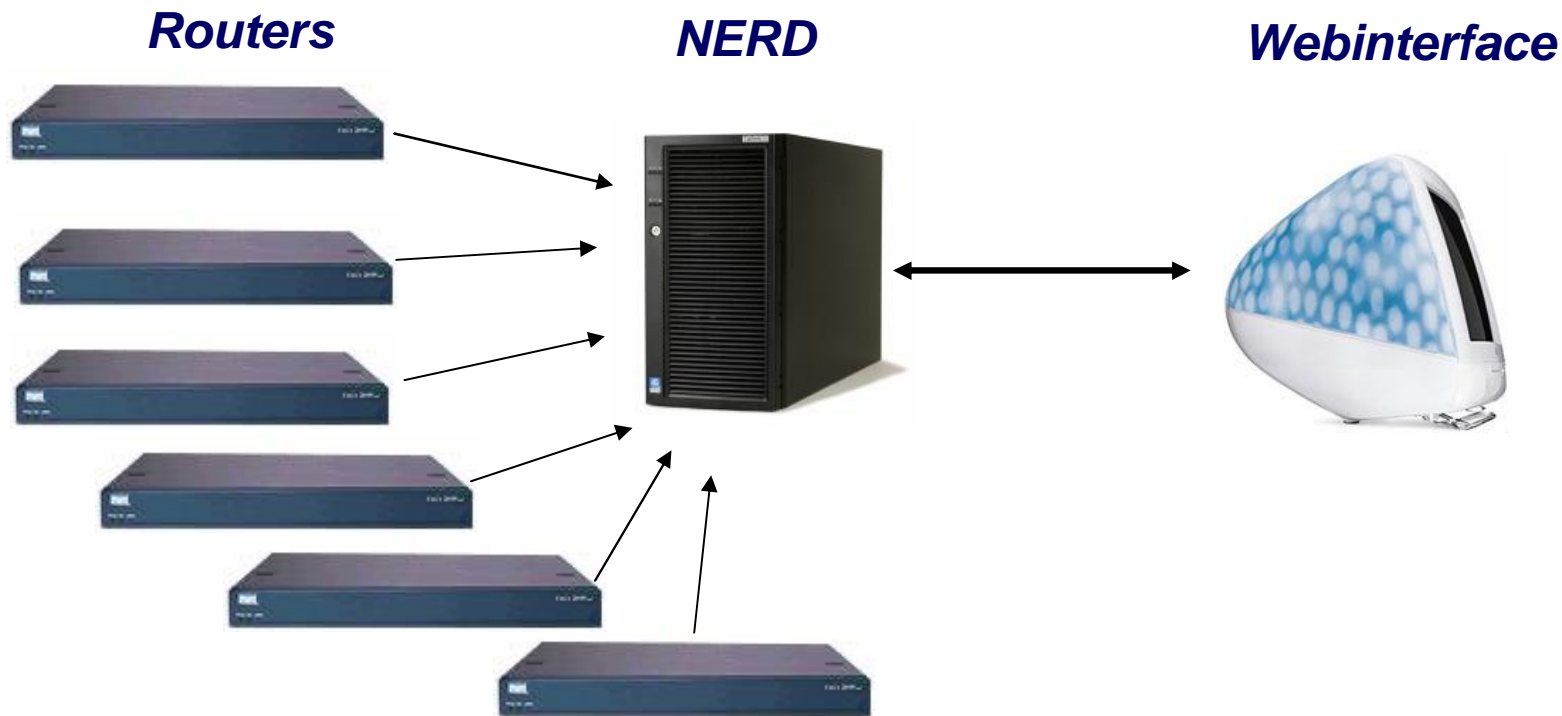
Discussion



- *What is N.E.R.D.?*

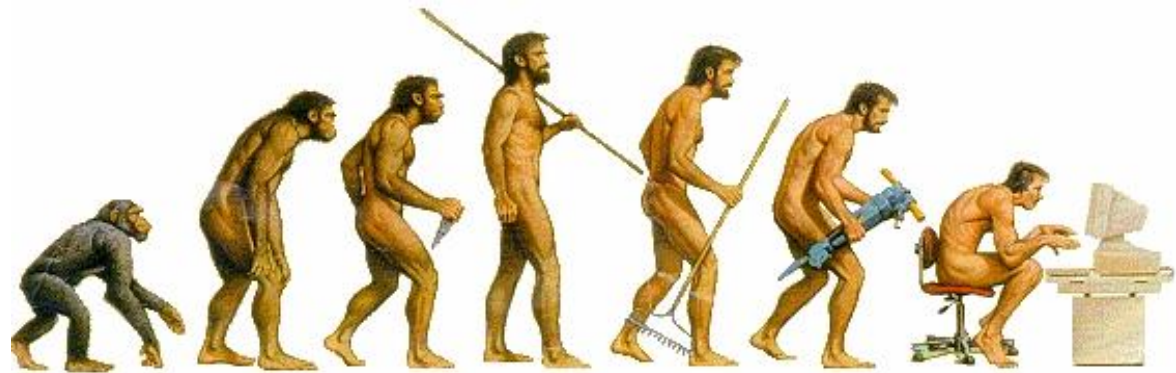
Network Emergency Responder & Detector (NERD):

- 4 Is a modular security monitoring tool that allows manual and automated analysis and response.
- 4 Collects and processes NetFlow information as input data



-History of N.E.R.D.

- 2002: SURFnet and TNO initiated a research project into DoS detection on the SURFnet network
- 2002: Prototype (NERD v0.1) finished, based on Caida's cflowd, flowtools, gnuplot, shell scripts and php code.
- 2003: Bugfixes in v0.1
- 2004: Design & development of NERD v0.5, removed third party tools by rewriting the daemon
- 2004: Design & development of NERD v1.0, bugfixes on daemon and new user interface
- '05/'06: Updates and bugfixes of NERD v1.1





Network Emergency Responder & Detector



Show all alarms from days ago, up to days ago.

The alarms between **2002-11-03** and **2002-11-04**

The query took approximately 0.001 seconds.

SYSLOG

- [Alarms](#)
- [Rules](#)
- [Messages](#)
- [PoP-map](#)
- [Search](#)

Destination IP address	Hostname	Flows per 5 minutes	Average packets per flow	Average bytes per flow	Average destination port	Starttime	Stoptime	Continuing
213.224.21.60	D5E0153C.kabel.telenet.be	36898	1	564	24116	2002-11-03 23:19:03	2002-11-03 23:24:03	0
194.204.175.120	z.lodz-r1.do.war-r2.tpnet.pl	36431	1	40	30801	2002-11-03 12:02:03	2002-11-03 12:07:03	0
217.98.48.98	planete.arx.pl	74354	1	40	32657	2002-11-03 11:47:04	2002-11-03 11:52:03	0
131.174.124.200	sun0irc.sci.kun.nl	68853	1	33	32730	2002-11-03 01:56:03	2002-11-03 02:06:02	0
66.67.226.217	hvc-66-67-226-217.hvc.rr.com	73641	1	575	12728	2002-11-03 01:26:02	2002-11-03 01:36:03	0

NETFLOW

- [Alarms](#)
- [Configuration](#)
- [Overall summary](#)
- [Analyse](#)
- [List](#)

NERD

- [System Info](#)

NERD

Network Emergency Responder & Detector

Alarms Analysis Settings ? ? ?

Load Cluster: [] Display: []

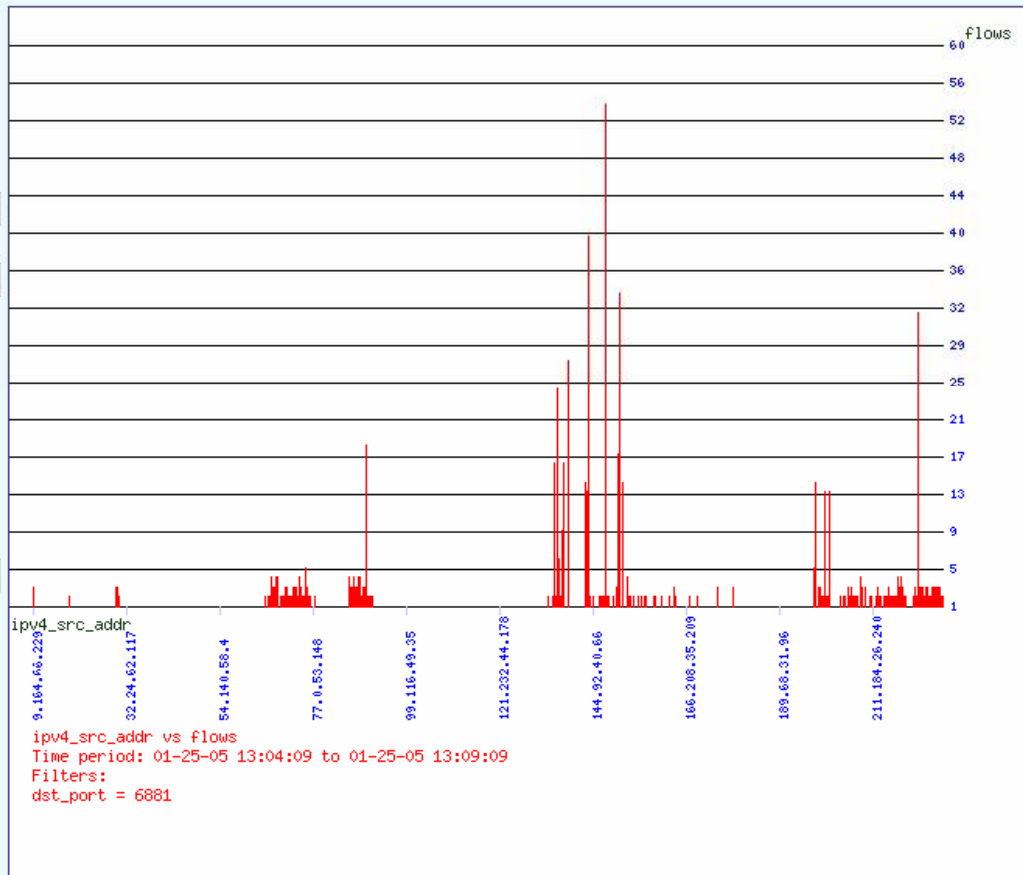
Group By: [] []

Load Filter: []

dst_port: [] = [] 6881

Start time: 1106654649.027981100 Stop time: 1106654949.009330300

Analyse



- **Background information**

Information required to understand the NERD basics:

- n Syslog
 - 4 The messages that are sent by the routers

- n Netflow
 - 4 Traffic information sent by the routers

- n Clusters & Filters
 - 4 Main type of analysis



Netflow is a Cisco standard for exchanging network traffic information.

Routers export netflow data, sending it to specific IP addresses

Data is collected by the routers either by processing all incoming packets or by sampling

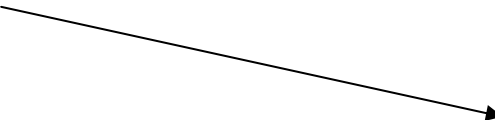
1 flow = an active connection between a source IP/port and a destination IP/port

IN_BYTES	MUL_DST_BYTES
IN_PKTS	LAST_SWITCHED
FLows	FIRST_SWITCHED
PROTOCOL	OUT_BYTES
SRC_TOS	OUT_PKTS
TCP_FLAGS	MIN_PKT_LNGTH
L4_SRC_PORT	MAX_PKT_LNGTH
IPV4_SRC_ADDR	IPV6_SRC_ADDR
SRC_MASK	IPV6_DST_ADDR
INPUT_SNMP	IPV6_SRC_MASK
L4_DST_PORT	IPV6_DST_MASK
IPV4_DST_ADDR	IPV6_FLOW_LABEL
DST_MASK	ICMP_TYPE
OUTPUT_SNMP	MUL_IGMP_TYPE
IPV4_NEXT_HOP	SAMPLING_INTERVAL
SRC_AS	SAMPLING_ALGORITHM
DST_AS	FLOW_ACTIVE_TIMEOUT
BGP_IPV4_NEXT_HOP	FLOW_INACTIVE_TIMEOUT
MUL_DST_PKTS	ENGINE_TYPE



Message with the following data:

- n Originating host (hostname/IP address)
- n Message priority level
- n Message itself
- n Timestamp



Emergency	(level 0)
Alert	(level 1)
Critical	(level 2)
Error	(level 3)
Warning	(level 4)
Notice	(level 5)
Info	(level 6)
Debug	(level 7)

Examples:

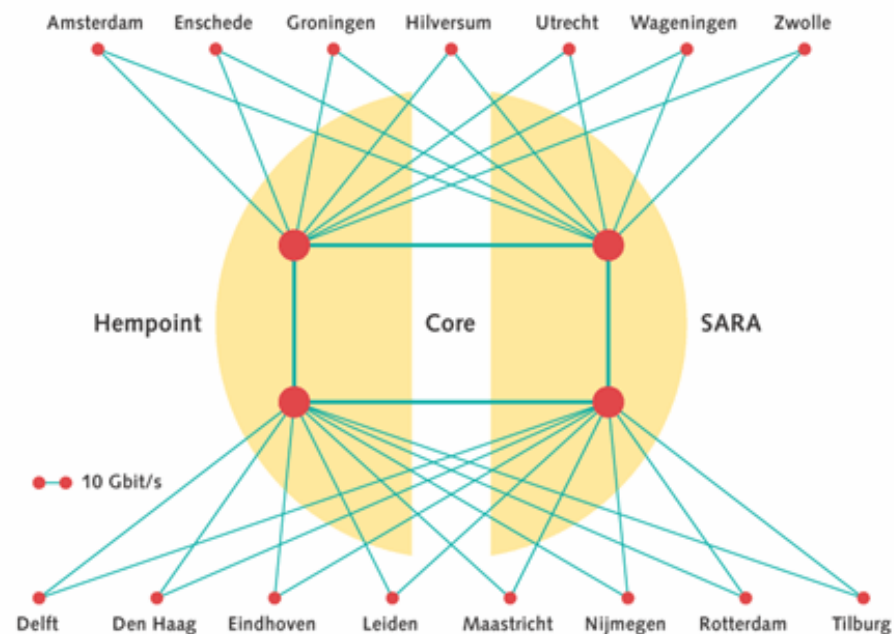
- n %SONET-4-ALARM: POS1/0: B2 BER below threshold, TC alarm cleared
- n %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial4/0/0, changed state to down
- n %SYS-2-PS_FAIL:Power supply 2 failed



-Netflow

Netflow is usually collected from all backbone routers

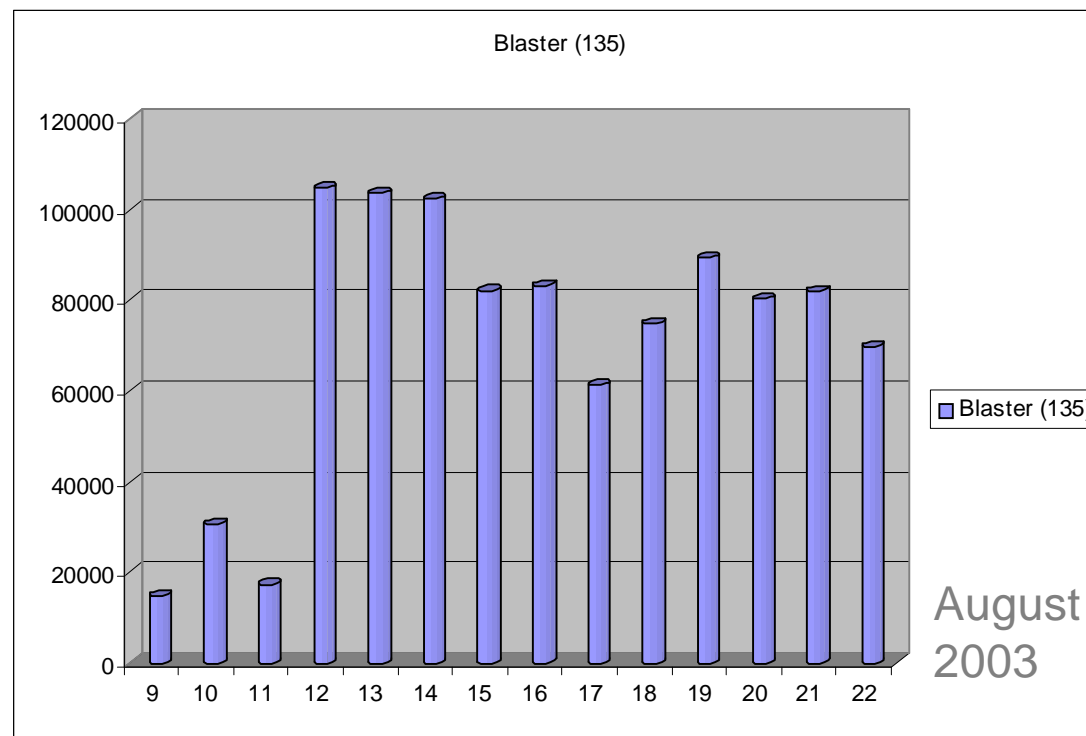
- n All flows will be seen twice, 'incoming' and 'outgoing'
- n Example image:



- Netflow works!

Netflow information can be used to detect attacks and incidents through the network

Not just DDoS attacks but worms as well:



- Clusters & Filters

N.E.R.D. is based on rules, if a user-defined rule is broken, an alarm will be generated.

A rule consists of the following fields:

- 4 Filter (Filters out unwanted information)
- 4 Cluster (Groups data by a shared property) (SQL Group By)
- 4 Limittype (Variable that triggers an alarm)
- 4 Limitvalue (The corresponding threshold value)

For example:

- 4 **Filter:** Destination port = 80, Number of packets = 1
- 4 **Cluster:** Destination IP address
- 4 **Limittype:** Flowcount
- 4 **Limitvalue:** 10000



- Clusters & Filters

Traffic info:

<i>Src IP add.</i>	<i>Dst IP add.</i>	<i>Src Port</i>	<i>Dst Port</i>	<i>Bytes</i>
10.0.0.1	192.168.0.25	31523	21	50
10.0.0.1	192.168.1.3	80	1042	80
10.0.0.30	192.168.0.25	1280	1042	60

Rule 1

- 4 No filter
- 4 Cluster on **Source IP address**,
- 4 Limit on **Flowcount**, maximum = 1

Result:

	(Source IP address)	(Flowcount)	
4	10.0.0.1	: 2	<- alarm
4	10.0.0.30	: 1	



- Clusters & Filters

Traffic info:

<i>Src IP add.</i>	<i>Dst IP add.</i>	<i>Src Port</i>	<i>Dst Port</i>	<i>Bytes</i>
10.0.0.1	192.168.0.25	31523	21	50
10.0.0.1	192.168.1.3	80	1042	80
10.0.0.30	192.168.0.25	1280	1042	60

Rule 2

- 4 No filter
- 4 Cluster on **Destination port**,
- 4 Limit on **Bytecount**, maximum = **100**

Result:

	(Destination Port)	:	(Bytecount)	
4	21	:	2	
4	1042	:	140	<- alarm



- Clusters & Filters

Traffic info:

<i>Src IP add.</i>	<i>Dst IP add.</i>	<i>Src Port</i>	<i>Dst Port</i>	<i>Bytes</i>
10.0.0.1	192.168.0.25	31523	21	50
10.0.0.1	192.168.1.3	80	1042	80
10.0.0.30	192.168.0.25	1280	1042	60

Rule 3

- 4 Filtering to accept only **Destination port > 1024**
- 4 Cluster on **Destination IP address + Destination port**,
- 4 Limit on **Flowcount**, maximum = 1

Result:

	(Destination IP / Port)	(Flowcount)
4	192.168.1.3 + 1042	: 1
4	192.168.0.25 + 1042	: 1

(No alarms)



Old NERD demo

4 (Syslog + Basic Netflow)

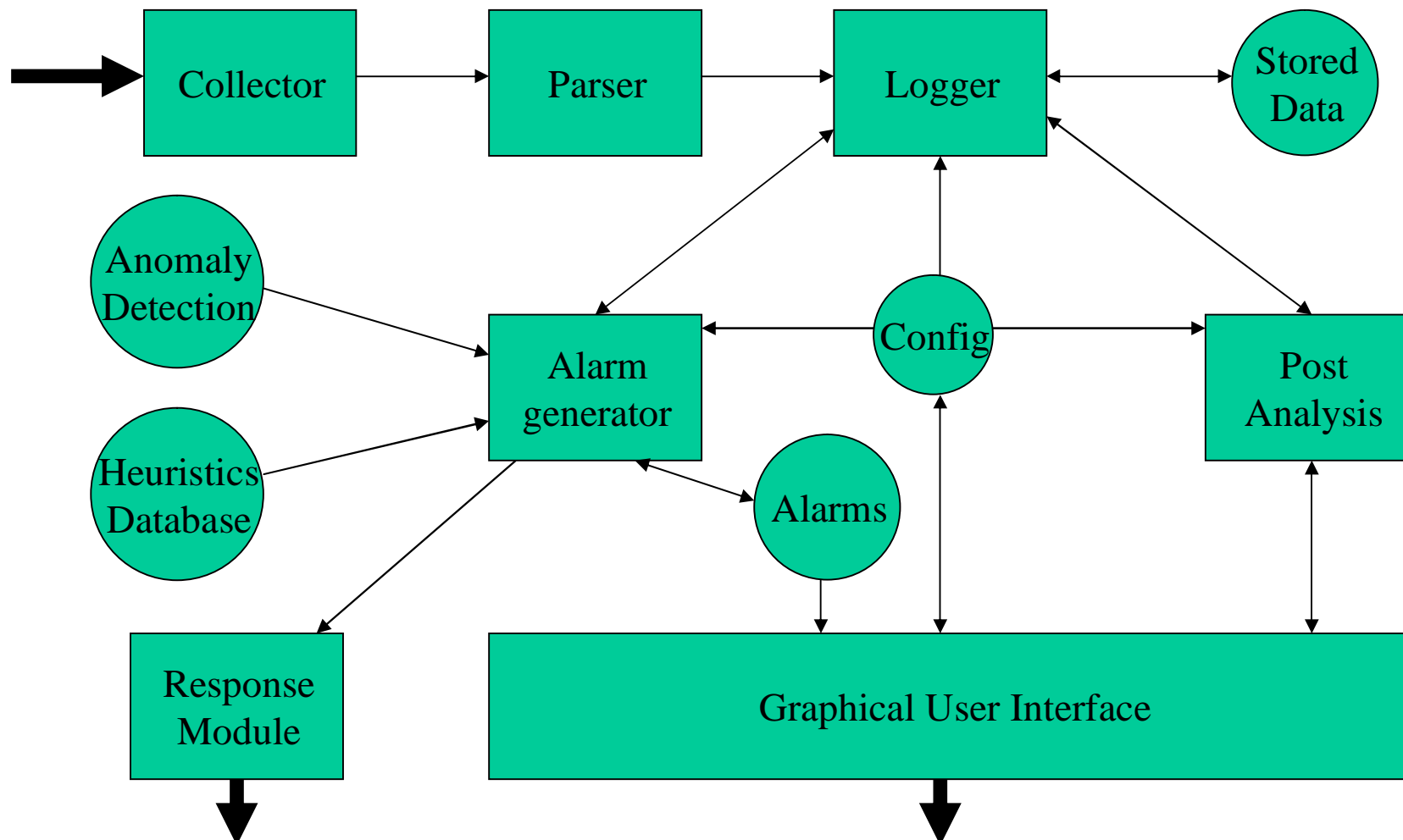
New NERD demo

4 (Advanced Netflow)

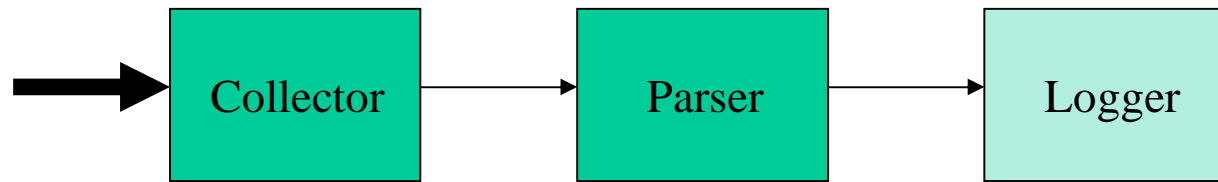


-The schematics:

Nerd v1.1 architecture



- Collector & Parser



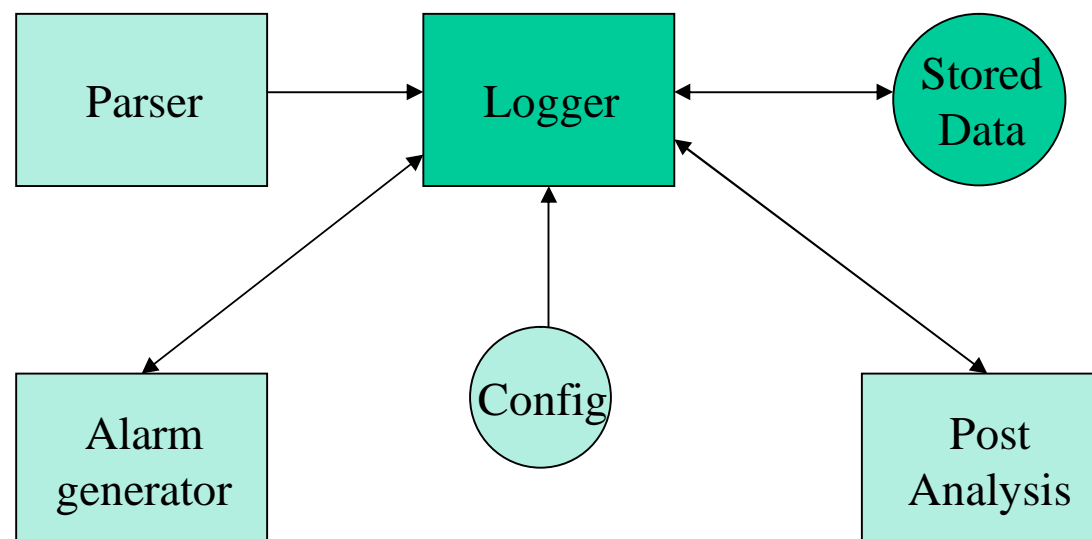
The collector is a process that receives the incoming flow information. This can be NetFlow, IPFix, etc.

Data is sent to the parser, which transforms NetFlow/IPFix/etc data into an internal format. This way, other processes only have to deal with the internal format, making a flexible approach of information input possible.

The parser sends the flow data (in the internal format) to the logger module.



- *Logger module*



The Logger stores the flow information in memory and disk. Memory provides faster access, which is useful for the Alarm generator as this module will request data regularly.

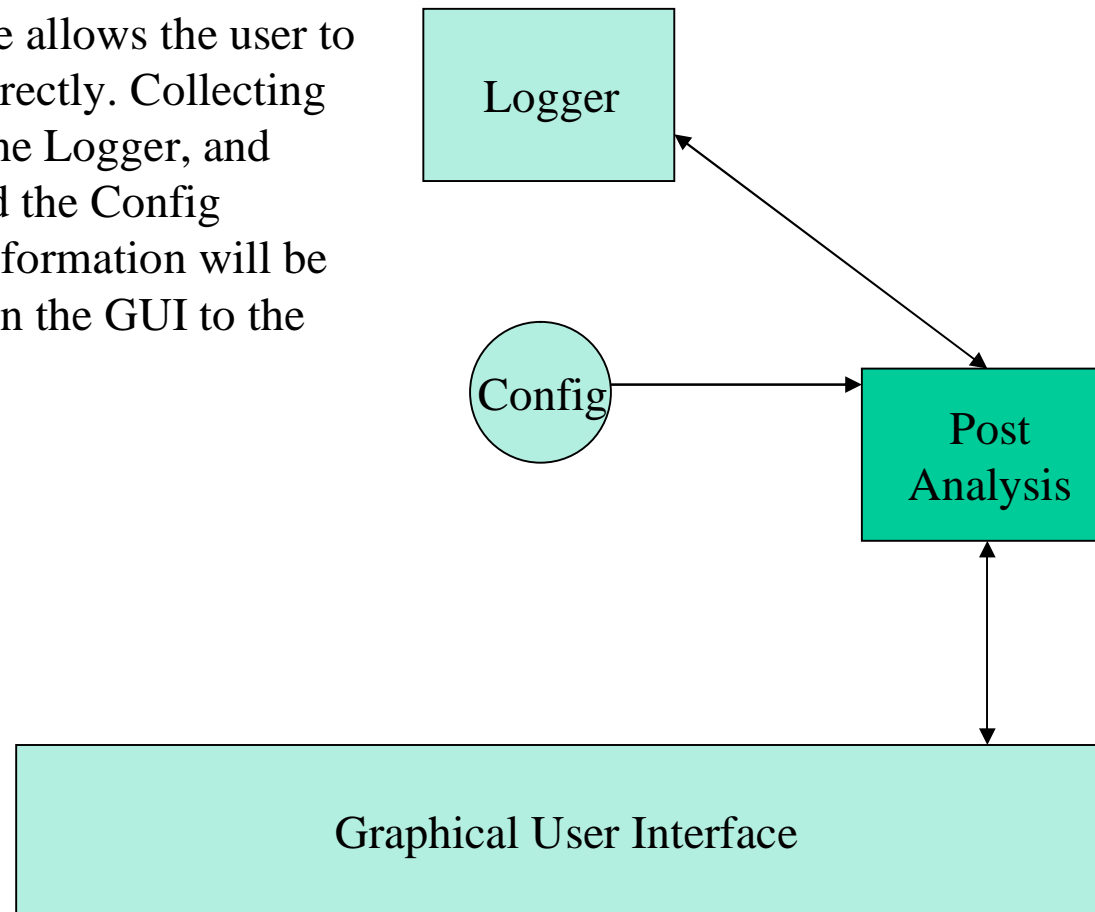
Disk provides slower access but has a higher capacity, allowing more data to be stored. This is useful for the Post analysis which will often request older data.

From the configuration database, settings are loaded like the amount of memory that can be used, etcetera.

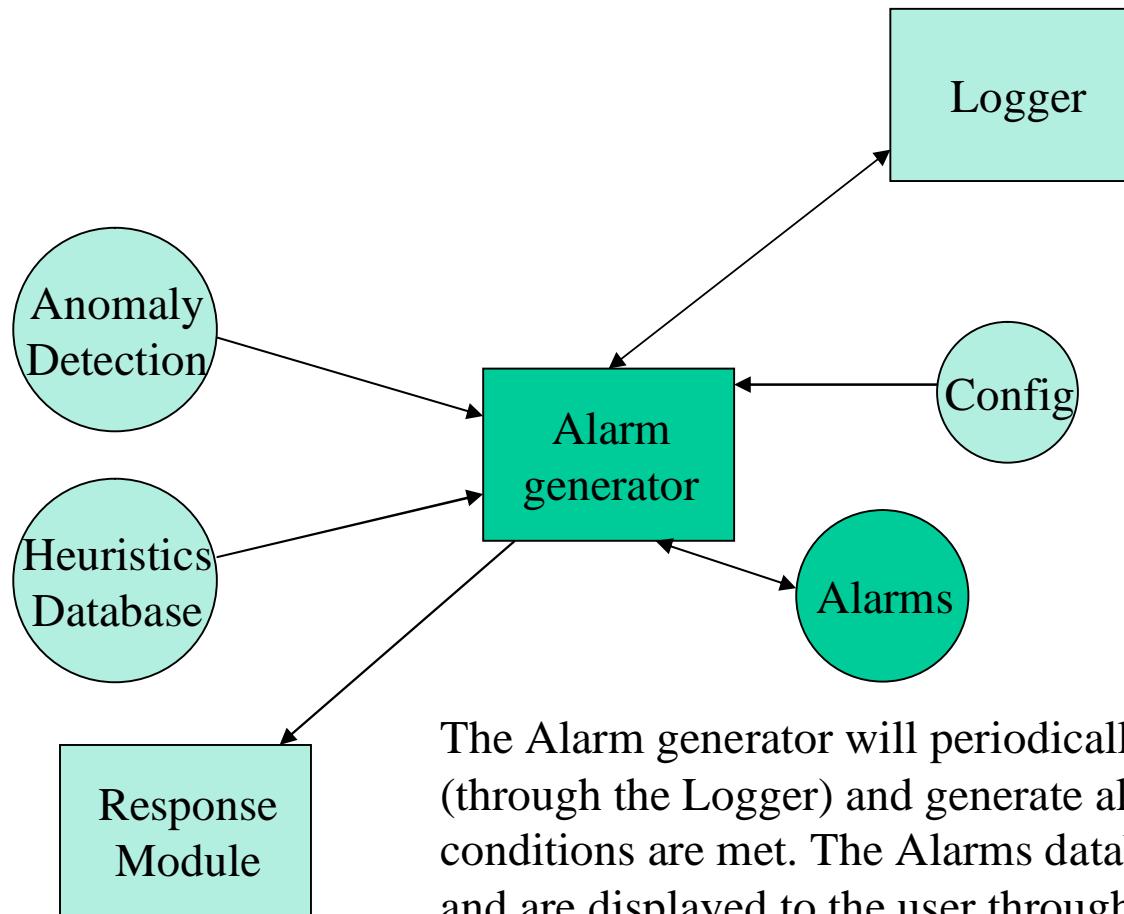


- *Post Analysis*

The Post Analysis module allows the user to analyse network traffic directly. Collecting the necessary data from the Logger, and settings from the GUI and the Config database, the requested information will be processed and displayed in the GUI to the user.



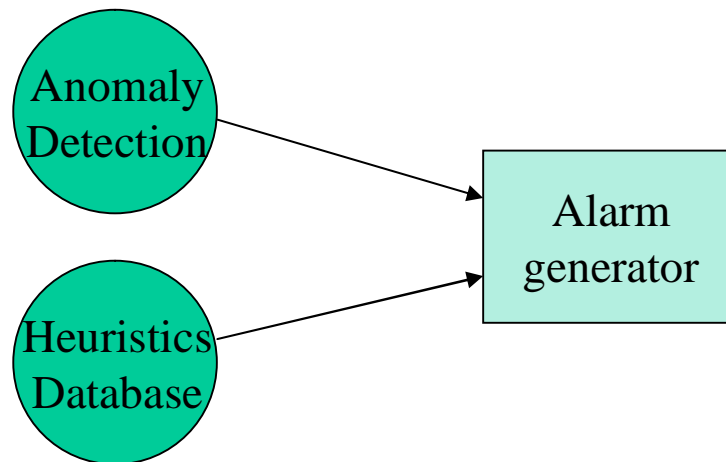
- Alarm generator



The Alarm generator will periodically examine recent flow data (through the Logger) and generate alarms if predefined conditions are met. The Alarms database will contain the alarms and are displayed to the user through the GUI.



-Anomaly Detection & Heuristics Database



To improve the incident detection of the Alarm generator, Anomaly detection can be employed. To detect anomalies, a pattern of regular traffic needs to be stored and updated.

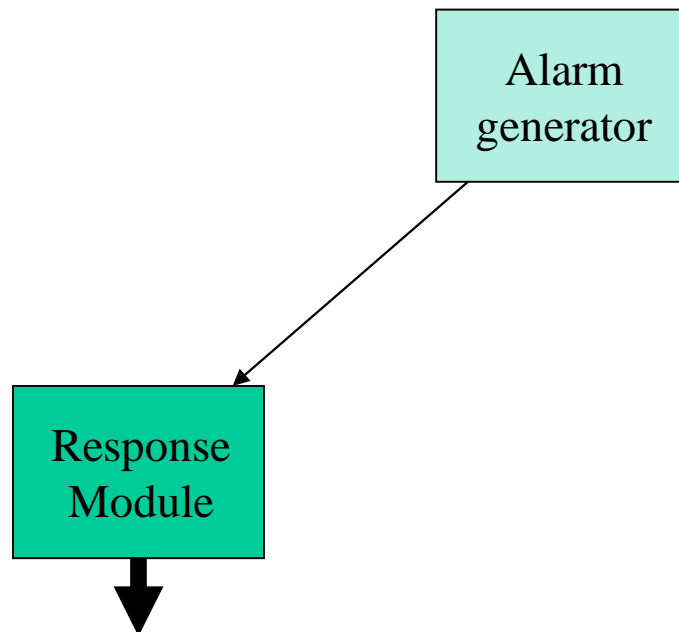
The Heuristics database aims at providing the user with detailed alarm information through an automatic alarm analysis. For example, indicate that alarm X is with 85% certainty a SYN-portscan, initiated from IP Y to subnet Z.



-Response Module

Besides displaying alarms, additional automatic responses can be used:

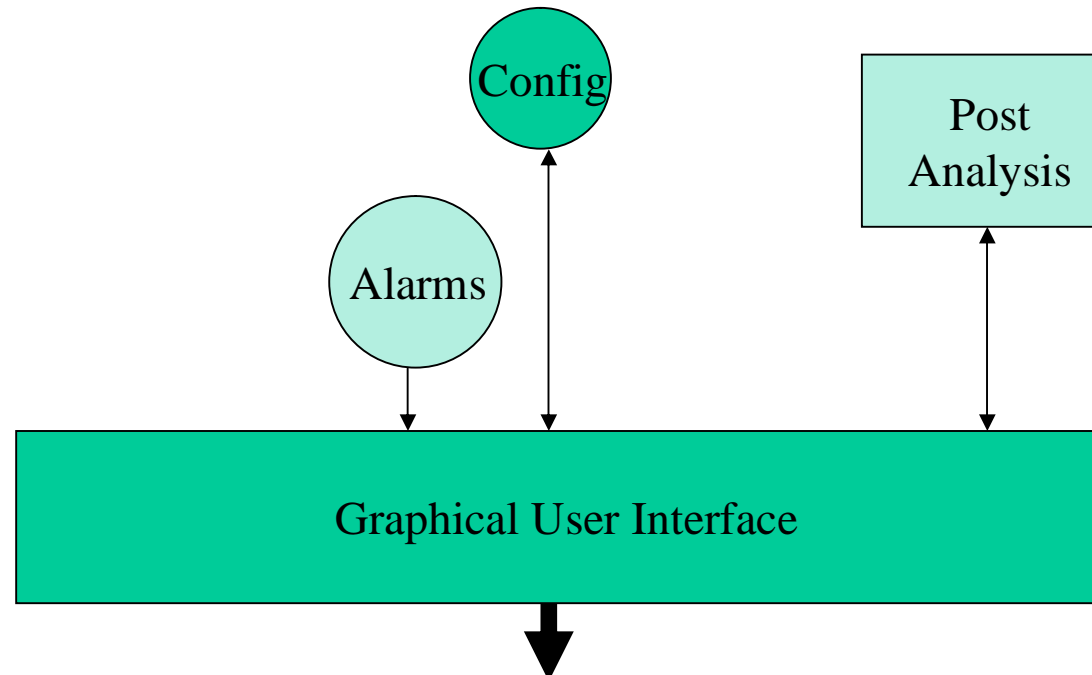
- Send a text message (SMS) to number X if an alarm occurs on subnet Y
- Update the Access Control List on a router to stop DDoS attack Z
- etc.



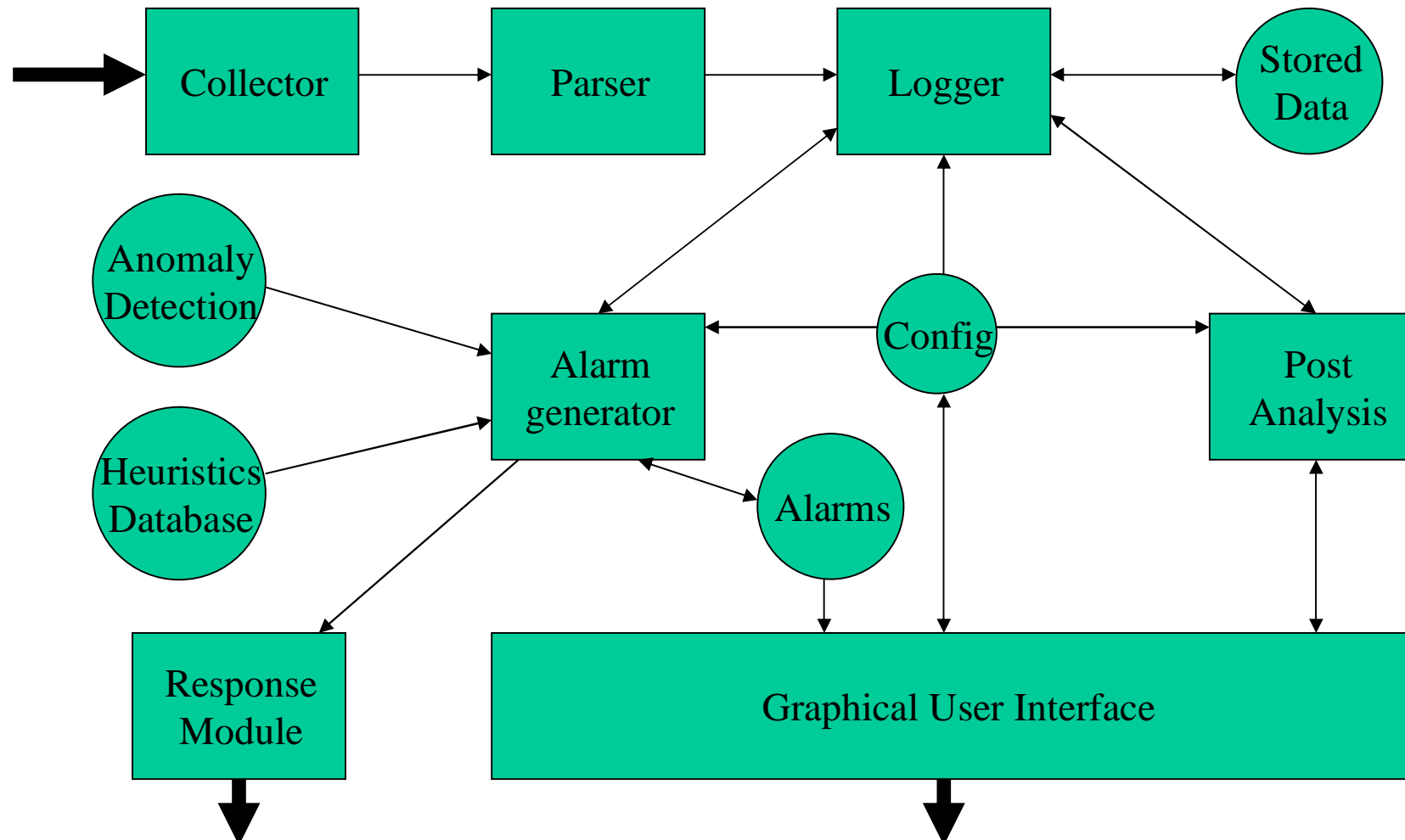
- Graphical User Interface

The GUI provides a front end to the user. All alarms are displayed here, and the user can make changes to the configuration of the monitoring system. The post analysis module allows the user to analyse flow information.

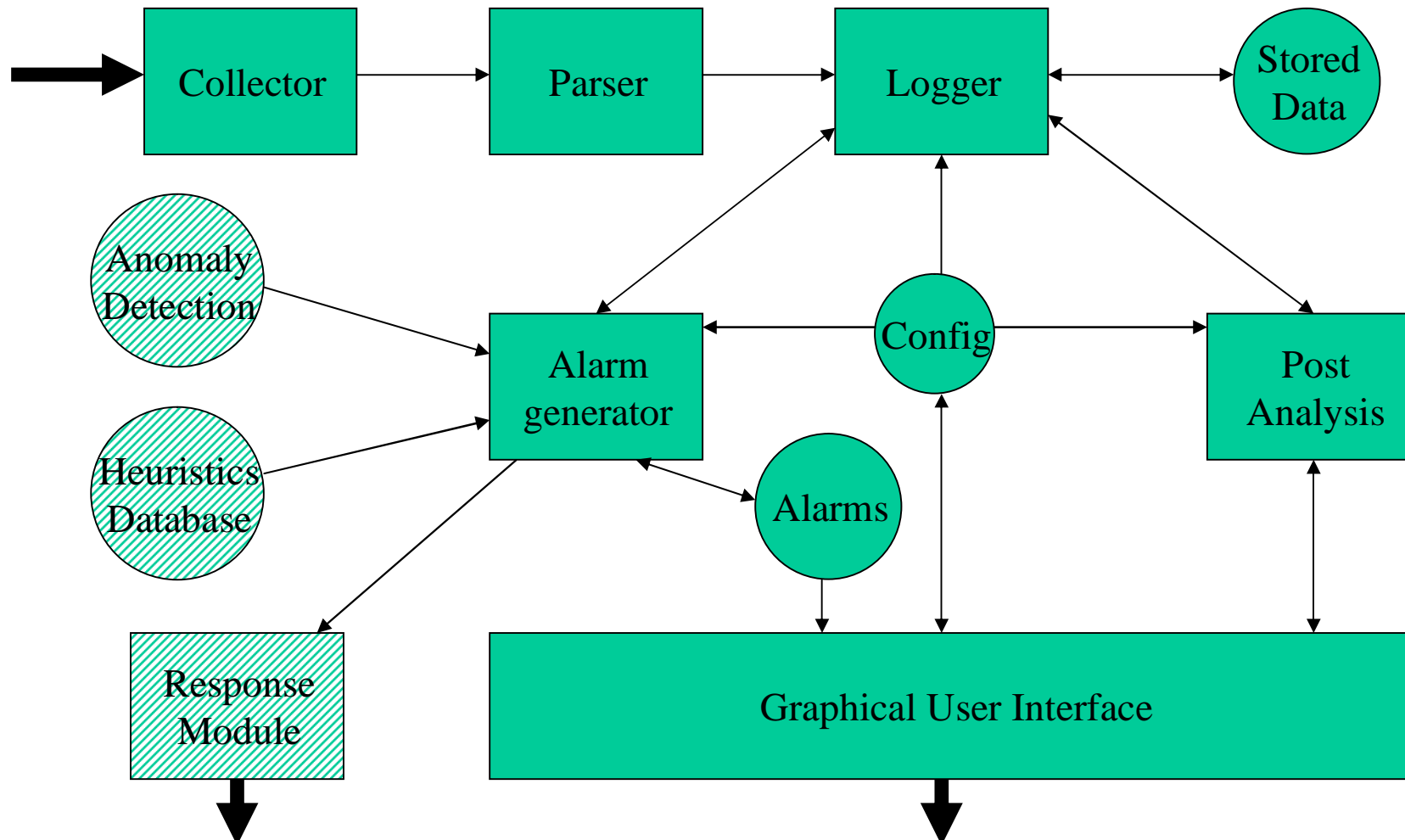
The configuration database contains all the configuration settings of the monitoring system, varying from GUI settings to alarm conditions.



- Overview again:



- Current status:



- Still TODO

Improve the analysis capability

- n Currently, alarms are triggered based on a threshold for flows, packets, octets, or NetFlow packets. We expect that additional “triggers” will prove useful. Testing of NERD in a large user community will tell what additional analyses might improve the tool. Currently we think of including statistical analysis. Furthermore, we want to include more types of graphs in the post-analysis.

Performance improvements

- n Due to the large amounts of flow data to be processed the performance of the system will be a point that needs continuous improvement. One of the ideas we have is to include performance filters in rules. For example some types of analysis do not require analyzing all data, but only a sample of the data.
- n A rule could for example include a filter: “sample rate = 100”. This would significantly speed up the analysis.

Usability

- n Based upon user experiences, the usability of the system can be improved.

Raw data monitoring

- n We expect that an integration of flow-based monitoring and raw-data monitoring is useful. Together with Cesnet and Uninett we are currently working in the LOBSTER project on an infrastructure for raw-data monitoring. The GN2 community is the primary target audience for this monitoring infrastructure.

Multiple users / User management

- n By providing a multiple user interface, customers could use the system to monitor their own network, which is a only a part of the total network that is monitored. User management would be required to create/delete/modify accounts through an easy to use interface.



- Still TODO

Self learning

- n In the first NERD prototype we have worked with a self-learning memory to limit the number of false alarms in real-time analysis. Such a self-learning memory can for example compensate for “normal” day/ night differences in network traffic for certain services. A generic version of this mechanism from the NERD prototype could be designed and implemented.

Automatic analysis

- n Once the tool triggers an alarm then for some alarms an automatic analysis can be initiated and results stored. This would limit the amount of processing when a user is using the system.

Automated response

- n Optionally, the NERD system can generate automatic responses in the event of an alarm; for example by sending e-mail, text messages or even enable filtering to stop an alarm altogether.

Communication system

- n A distributed model with master & slave types, can separate traffic-data storage from data analysis. This allows for more flexibility in the implementation of the system.
- n Communication with other monitoring services / NERD systems can aid in anomaly detection.

Employ lists of IP addresses that are treated differently

- 4 Suspected zombies
- 4 Special customer networks
- 4 etc.



- Discussion

