

```
<erno> hm. I've lost a machine.. literally *lost*.
It responds to ping, it works completely, I
just can't figure out where in my apartment
it is.
```

```
-- bash.org #5273
```

Notes on network monitoring,
by Oliver Gorwits

Part systems, part Perl

- 🔊 Monitoring what?
- 🔊 Shrink-wrapped solutions
- 🔊 Not Invented Here
- 🔊 Can't resist a bit of Perl
- 🔊 Pretty graphs

Monitoring what?

As we know, there are known knowns.

There are things we know we know.

We also know there are known unknowns.

That is to say we know there are some things we do not know.

But there are also unknown unknowns,

The ones we don't know we don't know.

*Donald Rumsfeld
Department of Defense news briefing
Feb. 12, 2002*

Monitoring what?

- 🔊 Free and Open (unknown unknowns)
- 🔊 Who is where on the network
- 🔊 What have they been up to?
- 🔊 What have my colleagues been up to?

Who is where?

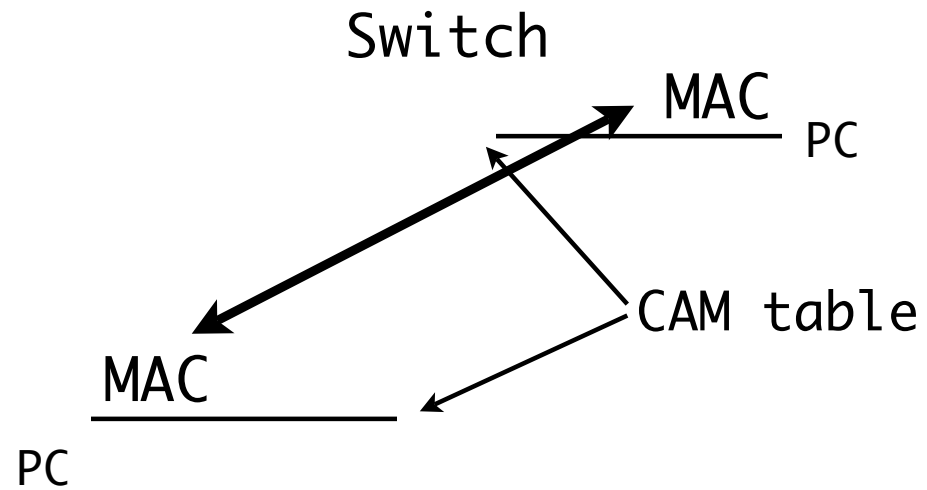
- 📌 MAC addresses

 - 📌 CAM tables - MAC to switch port

- 📌 IP addresses

 - 📌 ARP tables - IP to MAC

Who is where?



What have they been up to?

- 📌 Not sniffing - too much like work!
- 📌 Traffic levels
 - 📌 p2p, tut tut
- 📌 Problems
 - 📌 bad packets

Hive Mind, not

- 📌 ~6 humans, 100's devices, 1k's ports
- 📌 “There’s a problem *here*”
 - 📌 Where is *here* ?
 - 📌 Could be many places
- 📌 Too much changing for us all to track

Shrink-wrapped systems

- 📌 Black box rack-mount appliances
- 📌 Often OSS inside
- 📌 Manager-friendly web GUIs
- 📌 Might be fast and good
 - 📌 Might not be cheap

Net Mon tools

<http://www.slac.stanford.edu/xorg/nmtf/nmtf-tools.html>

Dirty little secret

- 🔊 Don't tell anyone, but...
- 🔊 No new developments in network monitoring in the last 20 years
- 🔊 I kid you not.

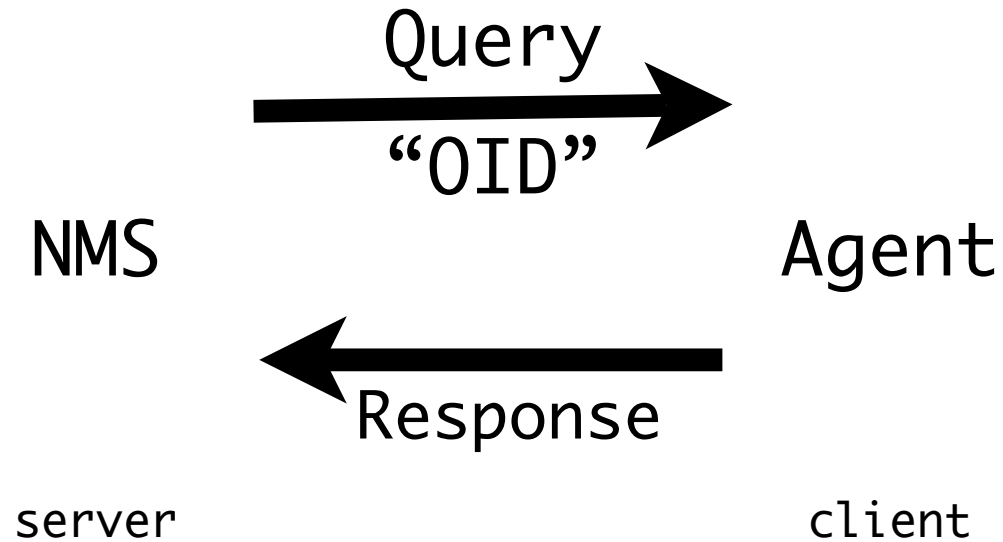
Monitoring 101

- 🔊 Send query to system
- 🔊 Get response
- 🔊 Act on response (optional)
- 🔊 er, that's it.

SNMP

- 🔊 Simple Network Management Protocol
- 🔊 Created in 1988
- 🔊 Not just for the network
 - 🔊 Printers, power supplies, Linux...

SNMP 101



Dirty little secret

- 📌 “Fancy” appliances just use SNMP
- 📌 Net-SNMP library
 - 📌 Open Source
 - 📌 Reference Implementation
 - 📌 Perl API

Stop. What is it that you really need?

- 🔊 Reports for management
- 🔊 Integration with existing systems
- 🔊 Traffic, errors, what else?
- 🔊 Visualization
- 🔊 History
- 🔊 Platforms (server, and network kit)

What we needed

- 📌 Separate web GUI, database, poller
 - 📌 Separation of concerns - security
 - 📌 Extending use of existing systems
- 📌 OIDs - Rumsfeld's unknown unknowns
- 📌 Graphs
- 📌 Linux

Monitoring what?

- 🔊 Free and Open (unknown unknowns)
- 🔊 Who is where on the network
- 🔊 What have they been up to?
- 🔊 What have my colleagues been up to?

Netdisco

- Written in Perl
- Uses SNMP::Info wrapper to Net-SNMP
- PostgreSQL database
- Distributed web, database, poller
- Answers the Who, What, Where
- Auto-discovery

Netdisco Demo

comms.oucs.ox.ac.uk/netdisco

What have they been up to?

- 📌 Traffic levels

 - 📌 p2p, tut tut

- 📌 Problems

 - 📌 bad packets

- 📌 Free and Open (unknown unknowns)

Not Invented Here

- 📌 MRTG - grand-daddy of monitoring
- 📌 RRDTool - grand-son of MRTG
- 📌 Cricket - RRDTool + poller

RRDTool

<http://oss.oetiker.ch/rrdtool/gallery/index.en.html>

RRDTool

- Round-robin Database
- C, with Perl API
 - Can be slow
- Binary storage - vulnerable
- Limiting
 - You only find this out later

Not Invented Here

 RTG - C + MySQL

 Cacti - PHP

 OpenNMS - Java schmava

Cacti/OpenNMS

http://cacti.net/get_image.php?image_id=43&x=1095&y=972&quality=90
<http://demo.opennms.org/opennms/>

Requirements

- 🔊 Quick to develop
- 🔊 Perl
- 🔊 SNMP polling
- 🔊 Fast storage
- 🔊 Very long history (for some data)
- 🔊 Graphs

SNMP::Effective

- Friendly wrapper for Net-SNMP
- Exposes asynchronous polling
 - Parallel execution with callback
- Get device list from Netdisco
- Wrap this in a sleep/wake loop

SNMP::Effective

```
1 use SNMP::Effective;
2
3 my $snmp = SNMP::Effective->new(
4     max_sessions    => $NUM_POLLERS,
5     master_timeout => $TIMEOUT_SECONDS,
6 );
7
8 $snmp->add(
9     dest_host => $ip,
10    callback => sub { store_data(@_) },
11    get      => [ 'sysDescr' ],
12 );
13 # lather, rinse, repeat
14
15 # retrieve data from all hosts
16 $snmp->execute;
```

YATG

- 🔊 YATG Ain't a Traffic Grapher
- 🔊 It's a wrapper for `SNMP::Effective`
- 🔊 Adds:
 - 🔊 Config (via `Config::Any`)
 - 🔊 (Remote) data storage
 - 🔊 Oh, and graphing

YATG Config

```
1 ---
2 yatg:
3     oids:
4         "sysUpTime":          [memcached]
5         "cpmCPUTotal5minRev": [stdout]
6         "ifHCInOctets":       [ifindex, rpc]
7         "ifHCOctets":         [ifindex, rpc]
8         "ifInErrors":         [ifindex, memcached, diff]
9         "ifOutErrors":        [ifindex, memcached, diff]
10        "ifAdminStatus":      [ifindex, memcached]
11        "ifOperStatus":       [ifindex, memcached]
12        "ifAlias":            [ifindex, memcached]
13        "dot1dStpPortState":  [ifindex, disk, "192.2.1.10", "!192.1.1.0/24"]

14    communities: [public, anotherone]
15    dbi_connect:
16        - "dbi:Pg:dbname=netdisco"
17        - netdisco
18        - "MYPASSWORD"
```

Tie::File

```
1 use Tie::File;
2
3 my $filename = "/var/tmp/output.dat";
4 my @array;
5
6 tie @array, 'Tie::File', $filename
7     or die "failed to open $filename: $!";
8
9 @array[13] = 'blah'; # line 14 of file becomes blah
10 print $array[42]; # display line 43 of the file
11 splice @array, 55, 1; # remove line 56 of the file
```

 Needs to scan whole file for separators

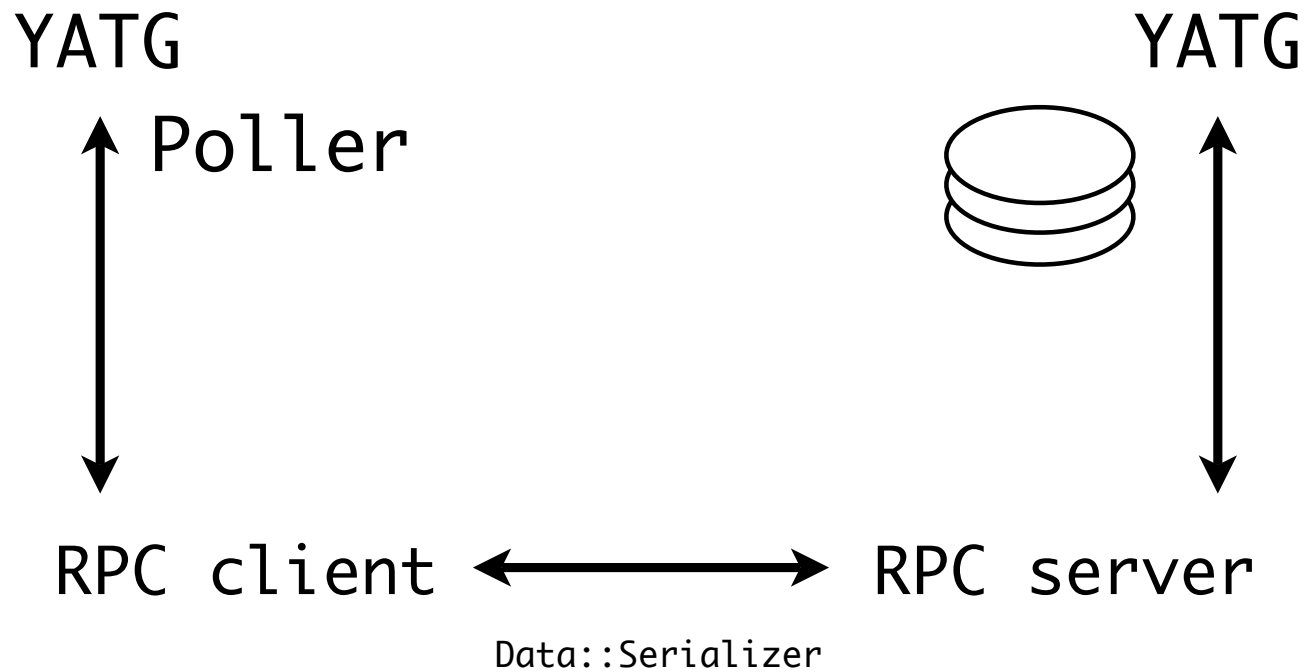
YATG Disk Storage

```
1 use Tie::File::FixedRecLen::Store;
2
3 tie @array, 'Tie::File::FixedRecLen::Store',
4           $file, record_length => 20
5           or die ...;
```

```
1 # pseudo-code
2 $filename =~ m/($time_stamp),($poll_interval)/;
```

- Create file if doesn't exist
- Line 0 data's timestamp is `$time_stamp`
- Line n data's timestamp is
 - `$time_stamp + ($poll_interval * n)`

RPC::Serialized



Nagios

- Generic service monitoring system
- HTTP, FTP, SMTP, DNS, DHCP, SNMP....
- Wide-open plugin interface
 - OK, WARNING, ERROR to STDOUT
- Web GUI, email alerts

Nagios

overlord.oucs.ox.ac.uk/nagios

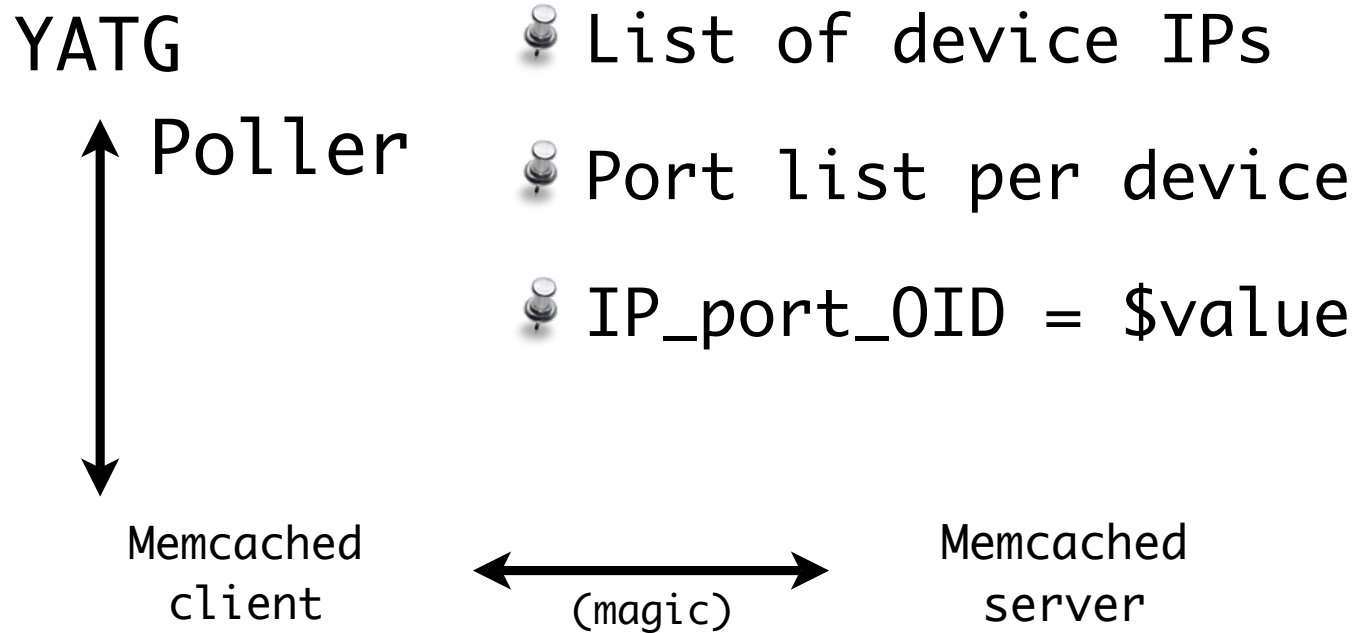
Feature creep

- Transient data, e.g. port state
- We have Nagios
 - A PITA to configure and manage
- Need a fast, simple, transient data storage facility

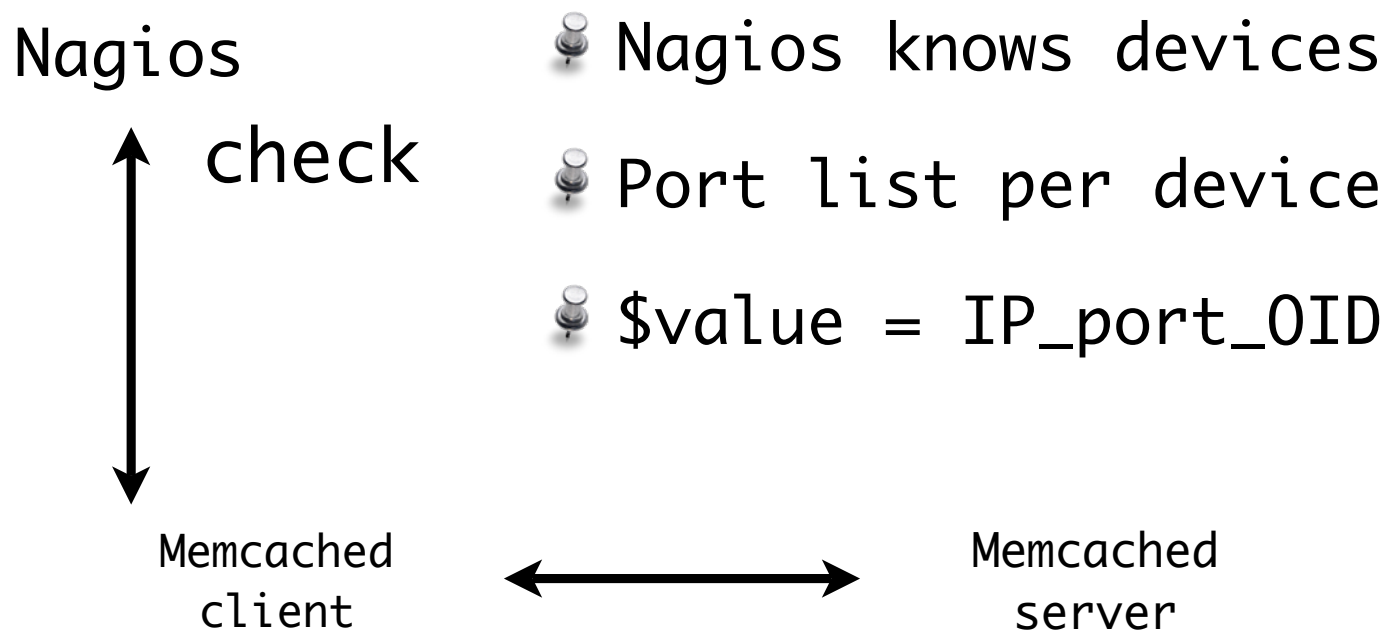
Memcached

- 📌 “Distributed memory object caching”
 - 📌 Designed to cache teh Intarweb
 - 📌 v. fast
 - 📌 Simple Perl API
 - 📌 Transient storage - data TTL

YATG Memcached Storage



Nagios, meet YATG



Crunch time

- 📌 ~7200 ports, 300+ devices
- 📌 5 minute polling interval
- 📌 24 seconds to poll these ports
 - 📌 11 OIDs
- 📌 9 seconds to store, over the network
 - 📌 Mix of disk and memcached
- 📌 Dual 2.8Gig Xeon, 4GB RAM






Eye candy

- 📌 YATG::Retrieve::RPC/Disk
 - 📌 Grab a time-slice of data from disk
- 📌 Chartdirector
 - 📌 Not Free nor Open,
 - 📌 but is cheap and quite brilliant!

YATG

comms.oucs.ox.ac.uk/netdisco

Invented Here

-  Netdisco
-  Nagios
-  Some Perl modules
-  YATG poller
-  Charthdirector

Fin.