Dr. NMS
or: How Facebook Learned to Stop Worrying and Love the Network

Jose Leitao [jleitao@fb.com] & David Rothera [drothera@fb.com]
We’ll be talking about

Facebook scale  FDN  Tales from the real world  This journey is 1% finished  Q&A
Facebook scale
as of December 2014

1.19 billion mobile
monthly active users

890 million daily active
users on average

1.39 billion monthly
active users

745 million mobile daily
active users on average
Facebook scale
as of December 2014

1.19 billion mobile monthly active users
890 million daily active users on average
1.39 billion monthly active users
745 million mobile daily active users on average

Approximately 82.4% of our daily active users are outside the US and Canada
What does that mean for the Facebook Network?
Lots of traffic and global footprint
Network traffic

Machine to machine

Machine to user
Engineers build robots, robots manage the network
Facebook Defined Networking
Facebook Defined Networking

Alert Manager Engine

- FBNet
- NetSonar
- Audit Framework
- Emitter
- One Detection
- NetNORAD
- syslog / SNMP traps
- ODS
- Megazord
- Vendors
- FBAR
- Tasks
- Poltergeist
- Carrier Maintenance
FBNet

The brains to help guide our robots run the network
Drain Services

The movers of traffic on the network devices
NetNORAD

Our Packet Loss detection system

© 2015 Facebook | Dublin | Credits: icon created by Pantelis Gkavos for https://thenounproject.com/term/radar/61238/; https://creativecommons.org/licenses/by/3.0/
Megazord

Our alarm correlation engine
Facebook Defined Networking

all components in action
Tales from the real world
Circuits @ scale

Manual approach

Hybrid approach
How is it now?
Fully automated
How is it now?
How is it now?

Notification from vendor

Parser gets data
How is it now?

Notification from vendor

Parser gets data

Task created
How is it now?

Notification from vendor

Parser gets data

Task created

Poltergeist calls drain

Poltergeist

Parser

Task created

Notification from vendor

Drain!
How is it now?

Notification from vendor
Parser gets data
Task created
Poltergeist calls drain
Circuits drained
How is it now?

- Notification from vendor
- Parser gets data
- Task created
- Poltergeist calls drain
- Maintenance ends
- Circuits drained
How is it now?

- Notification from vendor
  - Poltergeist calls drain
  - Task created
  - Parser gets data

- Drain!
  - Poltergeist calls undrain

- Undrain!
  - Circuits drained
  - Maintenance ends
  - Notification from vendor

- © 2015 Facebook | Dublin | Credits: robot icon created by Alexander Wiefel [link], explosion icon created by inconsmind [link], wrench by Musket [link], code icon by Darren Barone [link], check-list by Lubo Volkov [link], hands created by iconsmind [link] for [link], explosion icon created by inconsmind [link].
How is it now?

- Circuits are back in prod
- Poltergeist calls undrain
- Circuits drained
- Maintenance ends
- Task created
- Parser gets data
- Notification from vendor
- Poltergeist calls drain

What about fiber-eating sharks?
Seriously...
How is it now?
How is it now?

Link down alarms

Megazord groups alarms
How is it now?
How is it now?

- Link down alarms
- Megazord groups alarms
- Vendors logs event in OperDB
- Vendors checks FBNnet
How is it now?

- Link down alarms
- Megazord groups alarms
- Vendors logs event in OperDB
- Carrier is contacted with details of event
- Vendors checks FBNet
- Task created
How is it now?

Vendors checks FBNet

Vendors logs event in OperDB

Megazord groups alarms

Link down alarms

Carrier is contacted with details of event

Links come back [Monitoring period starts]

Task created
How is it now?

- Link down alarms
- Vendors logs event in OperDB
- Megazord groups alarms
- Carrier is contacted with details of event
- Links come back
- Task created
- Event is closed
- Monitoring period ends
- Monitoring period starts
Handover
The memory leak debacle

Free memory over time
How would this be solved with humans?
Lots of them + coffee
How is it now?
How is it now?

ODS detector for free memory goes below threshold

Alarm is generated
How is it now?

ODS detector for free memory goes below threshold

Alarm is generated

FBAR takes the alarm
How is it now?

ODS detector for free memory goes below threshold

Alarm is generated

FBAR takes the alarm

Remediation logic check redundancy - calls drainer
How is it now?

ODS detector for free memory goes below threshold

Alarm is generated

FBAR takes the alarm

Remediation logic check redundancy - calls drainer

Drainer!
How is it now?

ODS detector for free memory goes below threshold.

Alarm is generated.

FBAR takes the alarm.

Remediation logic check redundancy - calls drainer.

Drainer takes traffic from device.

How is it now?

- ODS detector for free memory goes below threshold
- Alarm is generated
- FBAR takes the alarm
- Remediation logic check redundancy - calls drainer
- Drainer takes traffic from device
- Active CPU is reloaded
How is it now?

- ODS detector for free memory goes below threshold
- Alarm is generated
- FBAR takes the alarm
- Remediation logic check redundancy - calls drainer
- Drainer takes traffic from device
- Drainer
- Standby CPU takes over
- Active CPU is reloaded
How is it now?

ODS detector for free memory goes below threshold

Alarm is generated

FBAR takes the alarm

Remediation logic check redundancy - calls drainer

Drainer takes traffic from device

Redundancy recovers [Active is reloaded]

Standby CPU takes over

Active CPU is reloaded

Drainer!
How is it now?

1. ODS detector for free memory goes below threshold.
2. Alarm is generated.
3. FBAR takes the alarm.
4. Remediation logic check redundancy - calls drainer.
5. Drainer takes traffic from device.
6. Standby CPU takes over.
7. Active CPU is reloaded.
8. Redundancy is restored.
9. Redundancy recovers [Active is reloaded].
How is it now?

ODS detector for free memory goes below threshold

Alarm is generated

FBAR takes the alarm

Remediation logic check redundancy - calls drainer

Redundancy is restored
Redundancy recovers [Active is reloaded]

Standby CPU takes over

Active CPU is reloaded

Drainer takes traffic from device
Facebook Defined Networking
all components in action

FBNet
NetSonar
Audit Framework
Emitter
One Detection
NetNORAD
syslog / SNMP traps

Alert Manager Engine

Megazord
Vendors
FBAR
Tasks
Poltergeist
Carrier Maintenance

ODS

So, in 30 days…

all components in action
So, in 30 days...
all components in action

Would receive and process over 3.37B notifications from networking devices, 0.99% resulting in alarms.
So, in 30 days...
all components in action

Would run ~750K times on networking alarms.
Automatically resolving 99.6% of them.
So, in 30 days...
all components in action

Would act on ~300 maintenances.

Carrier Maintenance
So, in 30 days…
all components in action

Would notify carriers about ~1100 transport events.
So, in 30 days...
all components in action

Would correlate different alarms into ~1200 unique master alarms.
Single on-call in charge of the whole network
Lessons Learned & Recommendations
1. Re-use existing code/tools when possible and when it makes sense.
8 Lessons Learned & Recommendations

2

Re-use existing code/tools when possible and when it makes sense.

Hacks quickly become important tools.
8 Lessons Learned & Recommendations

3

Re-use existing code/tools when possible and when it makes sense.

Hacks quickly become important tools.

Instrument / unit-test / document all the things.
8 Lessons Learned & Recommendations

4. Re-use existing code/tools when possible and when it makes sense.

Hacks quickly become important tools.

Instrument / unit-test / document all the things.

Poke for feedback often: if users don’t like the tool, they won’t use it.
8 Lessons Learned & Recommendations

5

- Re-use existing code/tools when possible and when it makes sense.
- Hacks quickly become important tools.
- Instrument / unit-test / document all the things.
- Networking devices don’t have powerful CPUs.
- Poke for feedback often: if users don’t like the tool, they won’t use it.
8 Lessons Learned & Recommendations

- Re-use existing code/tools when possible and when it makes sense.
- Hacks quickly become important tools.
- Instrument / unit-test / document all the things.
- The sooner the robots take over, the better.
- Networking devices don’t have powerful CPUs.
- Poke for feedback often: if users don’t like the tool, they won’t use it.
8 Lessons Learned & Recommendations

- Re-use existing code/tools when possible and when it makes sense.
- Hacks quickly become important tools.
- Networking devices don’t have powerful CPUs.
- Instrument / unit-test / document all the things.
- Talk is cheap, focus on impact.
- The sooner the robots take over, the better.
- Poke for feedback often: if users don’t like the tool, they won’t use it.
8 Lessons Learned & Recommendations

- **Done is better than perfect!**
- Talk is cheap, **focus on impact.**
- **The sooner** the robots take over, the better.
- Networking devices **don’t have powerful CPUs.**
- **Re-use** existing code/tools when possible and when it makes sense.
- Hacks quickly become **important tools.**
- Instrument / unit-test / document **all the things.**
- **Poke for feedback often:** if users don’t like the tool, they won’t use it.
- **Talk is cheap,** focus on impact.
What’s in the near future?

The journey is 1% finished

FBOSS / Wedge / sixpack operationalization improvements

Better visibility in the WDM space and correlation between the Optical / IP worlds

PCE

Continuous development of existing tools
WHAT WOULD YOU DO IF YOU WEREN'T AFRAID?