Managing Monitoring in Distributed Environments
Introduction

• Motivation
• Background
• Implementation
• Conclusions
Why Monitor?

“A distributed system is one that stops you from getting any work done when a machine you've never even heard of crashes.”

*Leslie Lamport*
Reliability

- In a replicated system, you only know when the last server fails.

- Sometimes you need to know that a system is down before the first user notices.
Why Monitor?

“Statistics are like bikinis. What they reveal is suggestive, but what they conceal is vital”

Aaron Levenstein
Performance Metrics

• Do we have “five 9s” uptime?

• Are there servers which regularly fall over?

• Are there patterns behind our service outages?

• Does Jim’s team manage their systems better than Susan’s?
Monitoring

- Regularly check if all services are working correctly
- Tell people if they’re not
Complications

Completeness

- Monitoring systems are rapidly taken for granted
- “I haven’t been paged - there can’t be anything wrong”
- Maintaining the monitoring system is often forgotten
- Easy to forget to add new services
- Omissions only noticed when things go wrong
- Although ... harder to forget when decommissioning!
Complications
Dependencies (and message storms)

- One small failure can have dramatic expenses
- Receiving 1,000 messages when a router fails can be both annoying and expensive
- Large scale monitoring configurations have to understand
  - Network topology
  - Service dependencies
- Modelling these is hard. Maintaining the model is harder still
Complications

fédéraux vers le Québec pourraient diminuer dès le lendemain d'une prise de pouvoir du Parti québécois, a laissé point de presse à Boucherville, hier. Au cœur d'une confrontation sur la question référendaire un ministre s'est défendu de vouloir mener une campagne de peur. Or, à ce sujet, il est largement vrai, les ambitions référendaires du PQ derniers jours, André Boisclair et François fiscal seraient versées au Québec :
evidence, être coupé subr c'est de faire sort réservé voir que pour le je ne... speak pas... français

Communication : Passive vs active

- Not everything you care about is network visible
  - Some things can’t be
  - Some things shouldn’t be

- An event occurring is as notable as a service outage

- Have to be able to handle
  - Services reporting their own state
  - Reporting of events (SNMP traps &c)
Complications

“Quis custodiet ipsos custodes?”

Juvenal

Image credit: Ley_photography http://www.flickr.com/photos/lostindevon/
Self monitoring

- Silence isn’t always golden
- Failure of a component of the monitoring system can hide everything
- Important that the monitoring system itself is robustly observed
Complexity and redundancy

• Satisfying all of these requirements leads to configurations that are
  • complex
  • fragile
  • hard to understand
  • full of redundant duplicated information

• Removing redundancy is highly beneficial. It lets us
  • ... make changes in one place instead of 20
  • ... avoid inconsistency between multiple copies
  • ... speed up configuration tasks
  • ... reduce errors!
Aside on redundancy

- System configuration is full of redundant information

- Take a web server configuration - information shared with
  - DNS
  - DHCP
  - Firewall
  - Certificate Server
  - Console Server
  - and your Monitoring System

- Removing redundancy dramatically reduces the potential for errors across all of these components
Lost in translation

• Very hard to handle raw configuration files

• Rapidly becomes an $O(n^2)$ problem (translate all of these configuration formats to all of these other formats)

• Instead, use a set of abstract configuration resources

• Translate from this to
  • Service configuration
  • Monitoring configuration
  • Everything else ...
Web server example

- This machine is an ssl capable web server

- It runs two virtual hosts
  - One http service on port 80, answering all requests, and redirecting to the https service
  - One https service on port 443, answering to ‘www.inf.ed.ac.uk’, serving documents from /var/www/html

- The https service should be given a signed certificate

- The web-service monitoring group is notified on failure
Implementation - Configuration

(other configuration systems are available)
Implementation - Monitoring
Example - Apache resources

#include <options/apache.h>
#include <options/apache-ssl.h>
#include <options/x509-client.h>

apache.vhosts                redir ssl
apache.vhostname_redir       _default_
apache.vhostverbatim_redir   Redirect / https://www.inf.ed.ac.uk

apache.vhostname_ssl         www.inf.ed.ac.uk
apache.vhostssl_ssl          true
apache.vhostroot_ssl         /var/www/html

apache.nagios_groups         nagios/web-service

X509_CERTIFICATE(ssl, /etc/pki/tls/certs)
Example - Apache configuration

ServerType standalone
ServerRoot /etc/httpd
LockFile /var/run/httpd.lock
PidFile /var/run/httpd.pid
ScoreBoardFile /logs/apache_runtime_status
Timeout 300
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15
MinSpareServers 5
MaxSpareServers 20
StartServers 8
MaxClients 150
MaxRequestsPerChild 100

LoadModule env_module /usr/lib/apache/mod_env.so
LoadModule config_log_module /usr/lib/apache/mod_log_config.so
LoadModule mime_module /usr/lib/apache/mod_mime.so
LoadModule negotiation_module /usr/lib/apache/mod_negotiation.so
LoadModule status_module /usr/lib/apache/mod_status.so
LoadModule includes_module /usr/lib/apache/mod_include.so
LoadModule autoindex_module /usr/lib/apache/mod_autoindex.so
LoadModule dir_module /usr/lib/apache/mod_dir.so
LoadModule cgi_module /usr/lib/apache/mod_cgi.so
LoadModule asis_module /usr/lib/apache/mod_asis.so
LoadModule action_module /usr/lib/apache/mod_actions.so
LoadModule userdir_module /usr/lib/apache/mod_userdir.so
LoadModule alias_module /usr/lib/apache/mod_alias.so
LoadModule access_module /usr/lib/apache/mod_access.so
LoadModule auth_module /usr/lib/apache/mod_auth.so
LoadModule digest_module /usr/lib/apache/mod_digest.so
LoadModule expires_module /usr/lib/apache/mod_expires.so
LoadModule setenvif_module /usr/lib/apache/mod_setenvif.so
LoadModule headers_module /usr/lib/apache/mod_headers.so

AccessFileName .htaccess
<Files ~ "\.ht">
Order allow,deny
Deny from all
Satisfy All
</Files>

UseCanonicalName On
TypesConfig /etc/mime.types
DefaultType text/html

ErrorLog logs/error_log
LogFormat "%h %l %u %t "%r" %>s %b "%{Referer}i" "%{User-Agent}i"
combined
LogLevel warn
CustomLog logs/access_log combined
SSLPassPhraseDialog builtin
SSLSessionCache shm:logs/ssl_scache(512000)
SSLSessionCacheTimeout 300
SSLMutex file:logs/ssl_mutex
SSLRandSeed startup builtin
SSLRandSeed connect builtin
SSLLog logs/ssl_engine_log
SSLLogLevel warn
AddType application/x-httpd-php .php
NameVirtualHost 129.215.165.30:80

<VirtualHost _default_:80>
ServerName _default_
Redirect / https://www.inf.ed.ac.uk
</VirtualHost>

<VirtualHost www.inf.ed.ac.uk:443>
ServerName www.inf.ed.ac.uk
SSLEngine On
SSLCertificateFile /etc/pki/tls/certs/ssl.crt
SSLCertificateKeyFile /etc/pki/tls/certs/ssl.key
SSLCertificateChainFile /etc/pki/tls/certs/ssl.crt
DocumentRoot /var/www/html
</VirtualHost>

Port 80
Listen 443
Listen 80

User apache
Group apache
ServerAdmin sxw
ServerName duffus.inf.ed.ac.uk
DocumentRoot /var/www/html

Options FollowSymLinks
AllowOverride None
</Directory>
Example - Apache monitoring

```plaintext
define command {
    command_line    $USER1$/check_http -I$HOSTADDRESS$ -p $ARG1$
    command_name    check_apacheconf_main
}
define command {
    command_line    $USER1$/check_http -I$ARG1$ -H$ARG2$ -p $ARG3$ -C$ARG4$
    command_name    check_apacheconf_cert
}
define command {
    command_line    $USER1$/check_http -I$ARG1$ -H$ARG2$ -p $ARG3$ -u $ARG4$ -S
    command_name    check_apacheconf_https
}
define command {
    command_line    $USER1$/check_http -I$ARG1$ -H$ARG2$ -p $ARG3$ -u $ARG4$
    command_name    check_apacheconf_http
}
define contactgroup {
    alias nagios/web-service
    contactgroup_name nagios/web-service
    members  sxw
}
define contact {
    use default-contact
    alias Simon Wilkinson
    contact_name  sxw
    email  sxw@inf.ed.ac.uk
}
define hostgroup {
    alias DICE/Web/Servers
    hostgroup_name DICEWebServers
}
define host {
    use default-host
    address 129.215.165.30
    contact_groups nagios/web-service
    host_name  duffus
    hostgroups  DICEWebServers
}
define service {
    use default-service
    active_checks_enabled  0
    check_command  check_self_active
    contact_groups  nagios/web-service
    host_name  duffus
    passive_checks_enabled  1
    service_description  Profile Translation
}
define service {
    use default-service
    check_command  check_apache_main!80
    contact_groups  nagios/web-service
    host_name  duffus
    service_description  Apache
}
define service {
    use default-service
    check_command  check_apache_http!129.215.165.30!www.inf.ed.ac.uk!80!/
    contact_groups  nagios/web-service
    host_name  duffus
    service_description  Apache www.inf.ed.ac.uk:80 HTTP
}
define service {
    use default-service
    contact_groups  nagios/web-service
    host_name  duffus
    service_description  Apache www.inf.ed.ac.uk:443 HTTPS
}
define service {
    use default-service
    contact_groups  nagios/web-service
    host_name  duffus
    service_description  Apache www.inf.ed.ac.uk:443 Certificate
}
define service {
    use default-service
    contact_groups  nagios/web-service
    host_name  duffus
    service_description  Apache www.inf.ed.ac.uk:443 Certificate
}
```

Large
Translation

• Have to translate our resources into application specific configuration

• Traditionally LCFG has ‘components’ which translate a machine’s resources into a local configuration

• Monitoring introduces more complexity
  • The monitoring configuration is made up of resources from many different machines
  • Introduce translators which convert machine resource fragments into monitoring configuration descriptions
Spanning Maps

• LCFG models configuration on a per host basis

• A publish/subscribe model exists for sharing information between hosts

• We call that model “spanning maps”

• We use it a lot: DHCP, firewall, Kerberos, X509

• Monitoring extends this, so it can see more, quicker
Monitoring Framework

- Monitoring system agnostic translation framework
  - Spanning maps tell it which components to monitor
  - Fetch resources for that component from database
  - Run resources through translator
  - Extract user/group information from user database
  - Combine translator results into configuration file

- All implemented in OO perl
  - Translators are run time loaded perl modules, written to a defined interface
Caching

- Monitoring reconfiguration is costly

- Have to cache as much as possible

- Framework caches
  - Results from configuration database
  - Results from user database
  - Results from translators

- Only moves performs reconfigurations when required
Why so much?

• Why is that configuration so complex?

• Actually monitoring 4 services
  • The default httpd instance
  • The virtual host running on port 80
  • The SSL service running on port 443
  • The validity of the certificate

• These all have dependencies
Dependencies

- Apache server
  - Redirect server
  - SSL server
    - SSL certificate
Redundant Servers and Clusters

- Modelling clusters is important
  - I really care if more than 2 of my KDCs are down
  - My LDAP service is dependent on there being a KDC available

- Nagios makes this hard, the framework makes it simple

```
kerberos.nagios_cluster       INF.ED.AC.UK_KDC
openldap.nagios_dependency    INF.ED.AC.UK_KDC
```
Notifications

- Notification storms are still a danger
- Systems tend to be maintained by more than one person
- Presence enables us to tell who is available
- Escalation lets us shout louder and wider
- Jabber is used for presence, and initial notification
- Escalations are by email
More on the Jabber bot

• Written in the python Twisted async framework

• Bot is permanently connected to our Jabber server, receives notifications by Unix socket

• Maintains state information for all of its buddies
  • Initially only notifies you if you’re ‘available’
  • Then notifies you if you’re online
  • Then emails you
Statistics

Using 2 monitoring machines, we monitor 159 services, across 42 hosts. The monitoring system has 5305 lines of configuration, all automatically generated and maintained.

(on 00:50 on 30th March 2008)

Image credit: williamhartz : http://www.flickr.com/photos/whartz/
Issues

• Only checking real world == configuration definition

• If you remove your web server from your configuration, the monitoring system won’t tell you it’s down

• Don’t have a good way of expressing
  • “There must be a www.inf.ed.ac.uk”
  • “There shall be at least 3 KDCs up at all times”
  • “Each site must have an AFS database server”

• Need a more descriptive configuration language to solve these problems
Futures

“Physician, heal thyself”

• Use monitoring events to repair problems
  • Bring up new systems when one goes down
  • Move volumes off AFS fileservers when one becomes too full
  • ...

• Lots of risks, but lots of potential
Conclusions

• A quick tour of some of the complexities of monitoring

• Examined how using a central configuration database can simplify these

• Described an implementation for LCFG and Nagios

• Looked at some possible ways of extending that implementation in the future
Questions?

This talk: http://www.dice.inf.ed.ac.uk/publications/

LCFG: http://www.lcfg.org/

Me: simon@sxw.org.uk