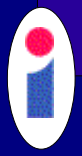
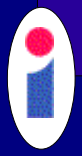
Simple Service Monitoring

Neil Brown
School of Informatics
University of Edinburgh
16/4/2008



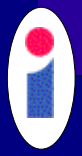
What's this talk about?

- Not a talk about Nagios or Monitoring. Simon's given talks on that and has the slides to prove it:
  http://www.dice.inf.ed.ac.uk/publications/UKUUG-LISA-2008/Monitoring.pdf
- Just going to cover the steps I took to add simple (Nagios) monitoring to the apache LCFG component using Simon's Translator tools.
- How to test the monitoring.
- How to use the monitoring (within our LCFG, Nagios environment)



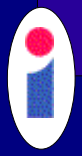
A Brief Overview

- Nagios is a system for monitoring systems. It is complicated and requires a lot of configuration.
- Simon setup http://nagios.inf.ed.ac.uk/ to monitor our systems, and it uses our LCFG configuration profiles to say what monitor.
- He has also written some Perl modules to make simple monitoring of a service easy. See:
  http://www.dice.inf.ed.ac.uk/units/infrastructure/Documentation/Monitoring/LCFG/Monitoring/Nagios/WritingTranslators.html
  which is a web version of
  perldoc LCFG::Monitoring::Nagios::WritingTranslators
- I used these Perl modules to add monitoring to apache.



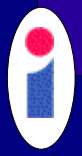
Why do you keep saying “simple”?

- Nagios is very powerful but complex. Even their own docs say it is hard to get up and running – but worth the effort.
- Fortunately for us, Simon has done a lot of the hard work, including hiding a lot of the gory details from us if we use the Perl modules he’s provided.
- In particular, LCFG::Monitoring::Nagios::Translators::SimpleCheck which, if you are lucky, will mean you only need to write a few lines of code for your Translator to work.
- We’re going to skip most of this detail, and basically work through Simon’s “Writing Translators” guide (link on previous slide).



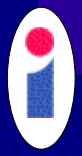
Can we get started yet?

- OK, so I wanted to add simple monitoring to the apache component. All I really want to know is, is there a web page being served by the host running the component.
- Nagios ships with some commands and plugins to check common networked services, look in /etc/nagios/commands.cfg and /usr/lib/nagios/plugins (on the nagios server).
- There is already a command to check HTTP, which does the simple check I needed. So all(!) I needed to do to take advantage of Simon's hard work, is create the Perl module:
  LCFG::Monitoring::Nagios::Translators::apache.pm
  to output the required check command.



Add Resources to the component

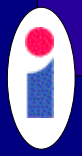
```
/* LCFG apache component : default resources */

#include "ngeneric-1.def"
#include "om-1.def"
#include "monitoring-1.def"

schema @SCHEMA@

conftmpl                /* location of http.conf template file */
config     conf/httpd.conf      /* config file */
serverroot /etc/httpd           /* server root directory */
httpd      /usr/sbin/httpd      /* the daemon */
lang       C                    /* the locale */

@startssl       %boolean      /* do we start the https server? */
startssl        false
```



Adding to the build

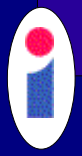
- By convention the component Translator goes in a subdirectory of the component source called “nagios”, and is called *component*.pm. We also create a suitable nagios/Makefile.PL.cin and later have this called by the main Makefile:

```
use ExtUtils::MakeMaker;

WriteMakefile(
      NAME      => 'LCFG::Monitoring::Translators::Nagios::@COMP@',
      ABSTRACT  => '@DESCR@',
      VERSION   => '@V@',
      AUTHOR    => '@AUTHOR@'
);
```



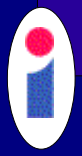
Update the Makefile

```
.PHONY: configure install clean translator
all: configure translator
[...]
configure: $(COMP) $(COMP).def $(COMP).pod $(NAME).$(MANSECT) \
           logrotate nagios/Makefile.PL
[...]
translator: nagios/Makefile
        cd nagios; make

nagios/Makefile: nagios/Makefile.PL
        cd nagios; perl Makefile.PL
[...]
install: configure translator
        @echo installing ...
        @cd nagios; make DESTDIR=$(PREFIX) pure_install
[...]
clean::
        @echo cleaning $(NAME) files ...
        @rm -f $(COMP) $(COMP).pod $(COMP).def $(NAME).$(MANSECT) logrotate
        @cd nagios; make clean
```



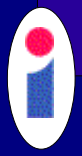
Update specfile

```
[...]
%install
rm -rf $RPM_BUILD_ROOT
@MAKE@ PREFIX=$RPM_BUILD_ROOT install
# Remove translator cruft
find %{buildroot} -type f -name .packlist -exec rm -f {} ';'
find %{buildroot} -type f -name '*.bs' -a -size 0 -exec rm -f {} ';'
find %{buildroot} -type d -depth -exec rmdir {} 2>/dev/null ';'
[...]
%package defaults-s@SCHEMA@
Summary: Default resources for @NAME@
[...]
%package nagios
Summary: Nagios translator for @NAME@
Group: @GROUP@/Translators/Nagios
BuildArch: noarch
%description nagios
Nagios translator for the LCFG @COMP@ component.
@LCFGCONFIGMSG@
%files nagios
%defattr(-,root,root)
%{perl_sitelib}/*

%clean
rm -rf $RPM_BUILD_ROOT
```



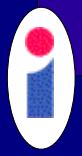
The apache Translator

```
package LCFG::Monitoring::Nagios::Translators::apache;

=head1 NAME

LCFG::Monitoring::Translators::apache - check an HTTPD server is running

=head1 DESCRIPTION

This translator produces configuration fragments to monitor an HTTPD server

It implements C<LCFG::Monitoring::Interfaces::Translator>, and requires the
resources documented in C<LCFG::Monitoring::Nagios::Translators::SimpleCheck>

Very minimal, just expects and 200 OK from the IP address.
=cut

use strict;
use warnings;
use base qw(LCFG::Monitoring::Nagios::Translators::SimpleCheck);
use LCFG::Monitoring::Nagios::Fragment::Command;

sub checkCommand {
  my ($self, $machine, $profile) = @_;

  return ("check_http");
}

sub serviceDescription {
  return ("apache");
}
1;
```



Testing Preparation

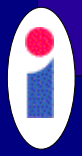
- Testing is important, because a broken Translator can break the whole monitoring system. But before you can test, you need to setup some things.
- Run ‘make [dev]rpm’, in the apache component source directory to create the usual RPMs. This includes an updated lcfg-apache-defaults-s1 RPM, which contains the new nagios default resources.
- Get this new defaults file installed on the profile server, eg update live/packages/live_testing_defaults.rpms and run updaterpms on the lcfg server.



Testing continued

- To the profile of the host you want to test against eg to scunner, add: !apache.nagios_groups mSET(neilb)
- Login to a machine with the dice/options/nagios_packages.h header file (or add it to your desktop machines profile). Don't use the nagios server (currently curlew).
- If necessary, do another make of the component and then 'cd' into the 'nagios' directory and run:

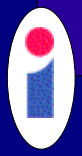
  lcfg-monitor-test --libs=./blib/lib apache scunner
- Check the output carefully.



Test Output

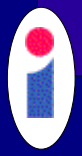
- You should get output that contains the following, with no error messages.

```
define service {
        use     default-service
        check_command   check_http
        contact_groups  neilb
        host_name       beezer
        service_description     apache
}
define contactgroup {
        alias   neilb
        contactgroup_name       neilb
        members neilb
}
define contact {
        use     default-contact
        alias   Neil Brown
        contact_name    neilb
        email   Neil.Brown@ed.ac.uk
}
```



Using your Translator - Installing

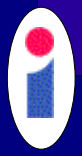
- Before you try to use your Translator to monitor your service, **it is important** that the nagios server has your Translator module installed before you try to use it.
- Possible RPM lists for your nagios translator RPM are (depending on if you are testing or if your component is LCFG or DICE level):
  - live/packages/live_testing_translators.rpms
  - core/packages/lcfg/lcfg-translators.rpms
  - core/packages/dice/dice_translators.rpms
- Then run updaterpms on nagios.inf and make sure the RPM installs.



Using your Translator – Defaults and Profiles

- Now that the Translator is installed on nagios.inf you need to ship the updated defaults file (now that it has the nagios resources) for the component (apache) to the LCFG server: live/packages/live_testing_defaults.rpms while testing, or core/packages/lcfg/lcfg-defaults.rpms to once you're happy.
- Once updaterpms has run on the server and the new defaults file has installed, you can try out the monitoring of your component.
- Add the following to machine's profile:

```
#include <dice/options/nagios_client.h>     /* turn on monitoring */
!nagios_client.manager    mSET(neilb)       /* can be a capability */
!nagios_client.components mADD(apache)      /* what to monitor */
!apache.nagios_groups     mSET(<%nagios_client.manager%>) /* who to alert */
```



That's it

- Assuming that http://nagios.inf.ed.ac.uk now shows your service being monitored, then that's it.
- Ah, one last thing that cropped up the other day. To **stop** monitoring your service/host, you need to remove the nagios resources from your host, and then make a whitespace change to the nagios.inf machine's profile (currently curlew) to trigger it to rebuild its configuration.
- Questions?
- Coffee?