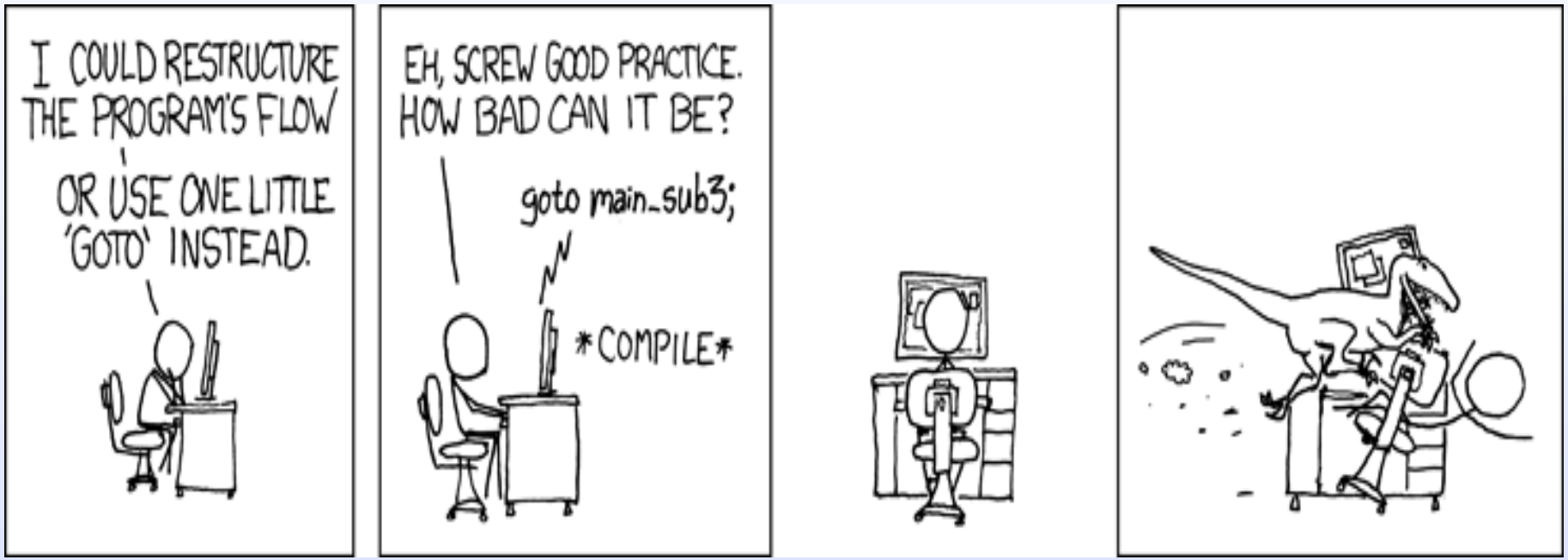
System Configuration : An end to hacky scripts?

Stephen Quinney <squinney@inf.ed.ac.uk>
School of Informatics, University of Edinburgh

The life of a Systems Administrator should never be dull!



http://xkcd.com/292/

From flickr.com (c) Buttonhead

From flickr.com
(C) *keng

From flickr.com (c) Pikesville

Example - resolv.conf

- Controls host name resolution
- Text file – designed to be simple
- Looks similar to:

```
nameserver 129.215.46.246
nameserver 129.215.64.240
search inf.ed.ac.uk
sortlist  129.215.46.0/255.255.255.0 129.215.144.0/255.255.255.0
```


Manual Approach

- One machine – Easy
- Several machines – Boring
- Many machines – Utterly Tedious

“I’ve got Perl* here and I’m not afraid to use it!”

*Substitute cool flavour-of-the-month scripting language

A “hacky” solution

- Recipe:
  1. Create list of all machines
  2. Write new config file
  3. Write short install script utilising scp

Problems

- What about uncontactable machines?
  - Need to handle timeouts
  - Need to keep a list of busy and dead
- What about machines belonging to others?
- What about new machines?

Further Issues

- What happens if you aren't a specialist?
  - Can you grok/edit sendmail configs?
- How can individual tasks be delegated?
- Typical solutions involve:
  - Templates
  - Version control system for config files

Push v Pull

- "Pushing" new config files to a host leads to various issues:
  - Typically involves manual intervention
  - Typically a "serial" approach - inefficient
  - Hard to handle dead and busy hosts
  - Harder to check successful installation
- Why not get the client to pull changes automatically and report back?

Extending the Example

- New requirement:
  - The order of the nameserver list must be randomised on a per-host basis.
- Simple solution:
  1. Generate all possible files.
  2. Make your script even hackier so it copies a random selection.

Still more problems!

- You might need to embed information which is host-specific.
- You might need information which is only on the physical host.
- Are your scripts **generic** and **reusable**?
- Do you share your scripts? Could anyone else use them?
- Are your scripts **documented**?

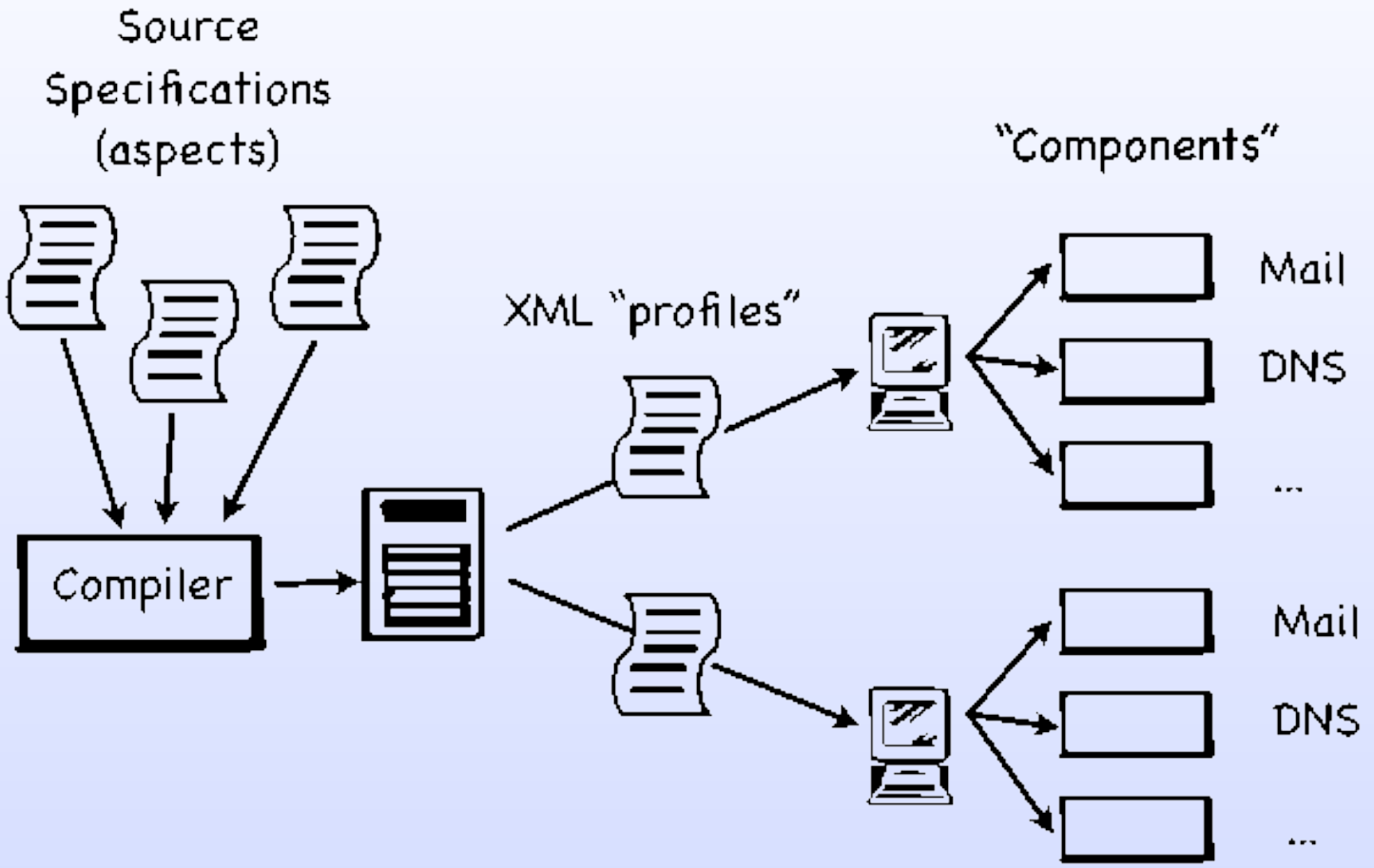
Managing Services

- A further extension of the "*managing file contents*" problem.
- Requirements:
  - Know **when** services should be restarted.
  - Know **how** to restart each service.
  - Only restart if the new config file is **valid**.
  - **Report** back success or failure of restart

LCFG – Overview

- Client/Server Architecture.
- Each client has a "*source profile*".
- Config files are built on the client using:
  - Data stored on the server as "*resources*".
  - Scripts and templates stored on client.
- Server processes the source profile and generates an XML representation.

LCFG Overview



LCFG – Component Overview

- Resources are logically grouped into "*components*".
- A profile consists of several components.
- Components have scripts which are based on a standard framework.
- Respond to a set of methods:
  - start, stop, restart, configure, run, etc..

LCFG – Client Overview

1. Pulls down the generated XML file.
2. Notices any resource changes.
3. Calls the "*configure*" method for affected components.

Back to resolv.conf

- Perl-based component (we also support shell).
- Need to sub-class LCFG::Component.
- Overrides the default Configure method.

lcfg-resolvconf - Code

```
sub Configure {
  my ( $self, $res ) = @_;

  my $status = LCFG::Template::Substitute(
                  $res->{template}{VALUE},
                  $res->{configfile}{VALUE},
                  4, $res );

  if ( ! defined $status ) {
    $self->LogMessage($@);
    $self->Fail( "update failed (see logfile)");
  }
  elsif ( $status == 1 ) {
    $self->LogMessage("successful update");
  }

  return;
}
```


lcfg-resolvconf - Resources

```
#include <lcfg/options/resolvconf.h>

resolvconf.nameservers      129.215.46.33\
                            129.215.46.246\
                            129.215.64.240

resolvconf.sortlist         129.215.46.0/255.255.255.0\
                            129.215.144.0/255.255.255.0\
                            129.215.41.0/255.255.255.0\
                            129.215.32.0/255.255.255.0

resolvconf.randomize        true
```

Sets up default resources & adds package

Component

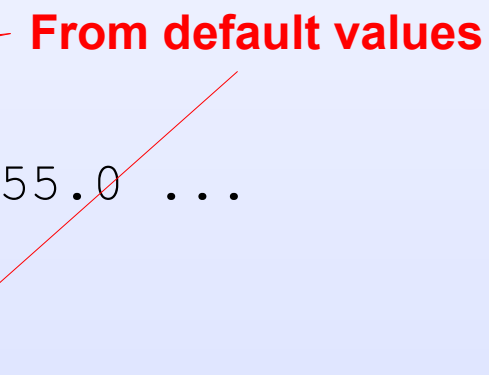
Resource name

lcfg-resolvconf - Template

```
[% FOR server IN nameservers.split('\s+') -%]
nameserver [% server %]
[% END -%]
[% IF search.length > 0 -%]
search [% search %]
[% END -%]
[% IF domain.length > 0 -%]
domain [% domain %]
[% END -%]
[% IF sortlist.length > 0 -%]
sortlist [% sortlist %]
[% END -%]
[% IF optstring.length > 0 -%]
options [% optstring %]
[% END -%]
```


lcfg-resolvconf - Output

```
nameserver 129.215.46.246
nameserver 129.215.64.240
nameserver 129.215.46.33
search inf.ed.ac.uk
domain inf.ed.ac.uk
sortlist 129.215.46.0/255.255.255.0 ...
129.215.144.0/255.255.255.0 ...
129.215.41.0/255.255.255.0 ...
129.215.32.0/255.255.255.0
options  ndots:1 timeout:5 attempts:2
```



Components for Services

- Make Configure call Restart on change
- Need to also override Start, Stop
- This might be as simple as:

```
Start() {
   /etc/init.d/openssh start
}

Stop() {
   /etc/init.d/openssh stop
}
```


Extra Benefits

- Resource validity checking
- Useful defaults – minimal effort needed
- Boot-time and configure-time sequencing
  - e.g. Add a user before using it for file owner
- Automated installer

Conclusions

- System configuration frameworks make life less dull!
- Provides the ability to:
  - Manage change automatically
  - Stop focussing on the implementation
  - Start thinking about the intentions
- System management moves to a higher level

Where To Now?

- http://www.lcfg.org/
- http://wiki.lcfg.org/
- info@lcfg.org
- SAGE System Configuration booklet - http://www.sage.org/pubs/14_sysconfig/
- SAGE LCFG booklet – coming soon!