Automating IPv6 Deployments

Ivan Pepelnjak (ip@ipSpace.net)
Network Architect

ipSpace.net AG

Who is Ivan Pepelnjak (@ioshints)

Past

- Kernel programmer, network OS and web developer
- Sysadmin, database admin, network engineer, CCIE
- Trainer, course developer, curriculum architect
- Team lead, CTO, business owner

Present

- Network architect, consultant, blogger, webinar and book author

Focus

- Network automation and Software Defined Networking
- Large-scale data centers, clouds and network virtualization
- Scalable application design
- Core IP routing/MPLS, IPv6, VPN

CISCO CCIE EMERITUS

vmware vEXPERT

Educate

↓

Research

↓

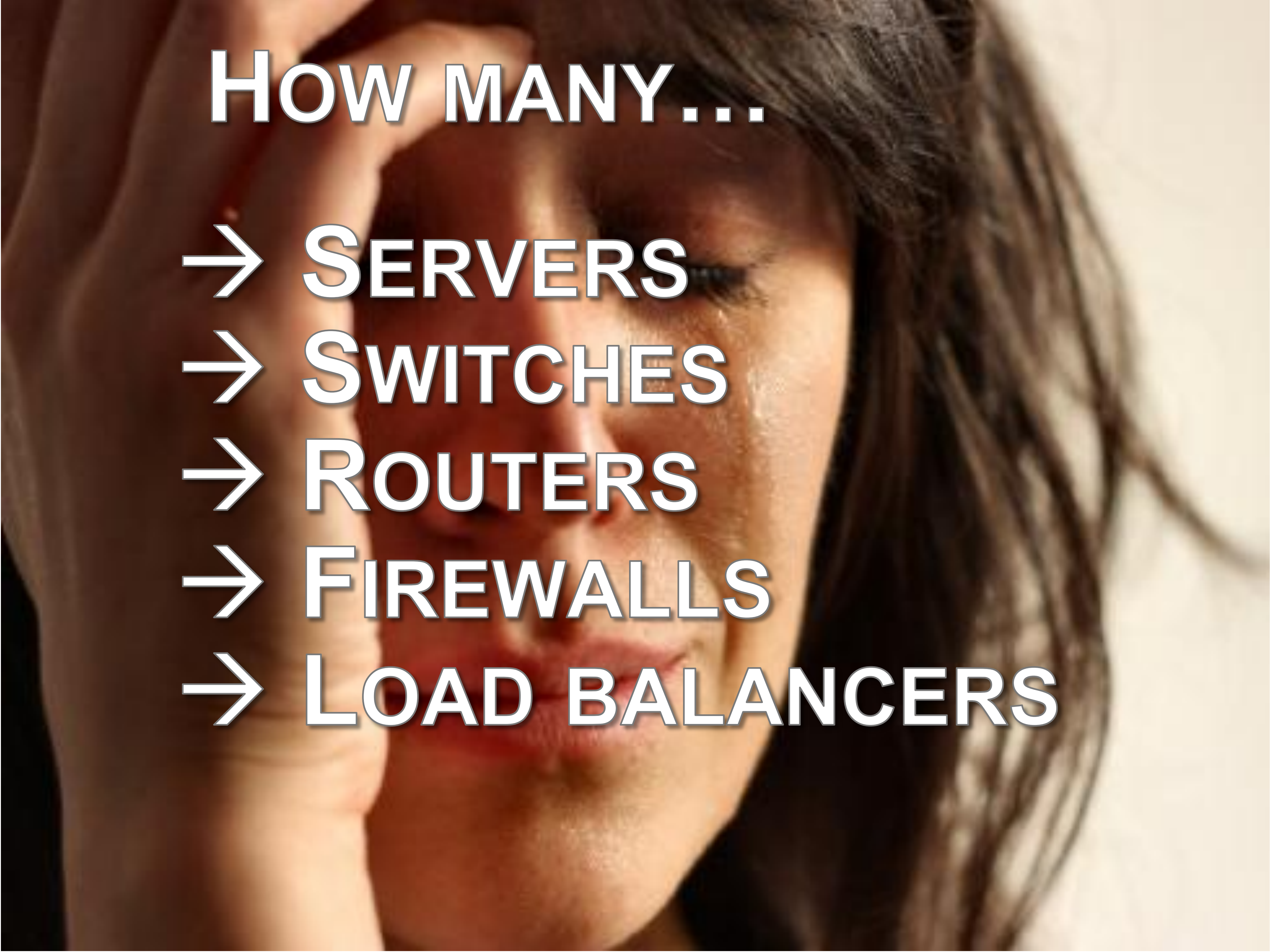
Design

↓

Test

↓

Deploy



How many…

→ Servers
→ Switches
→ Routers
→ Firewalls
→ Load balancers

It’s Utterly Boring

IPv6 configuration is very similar to IPv4 configuration

- Slightly different commands and caveats
- Different addresses
- Deploying IPv6 is boring…
- … and boredom results in mistakes

```
interface Loopback0
 ip address 10.0.1.1 …
 ip ospf 1 area 0
```

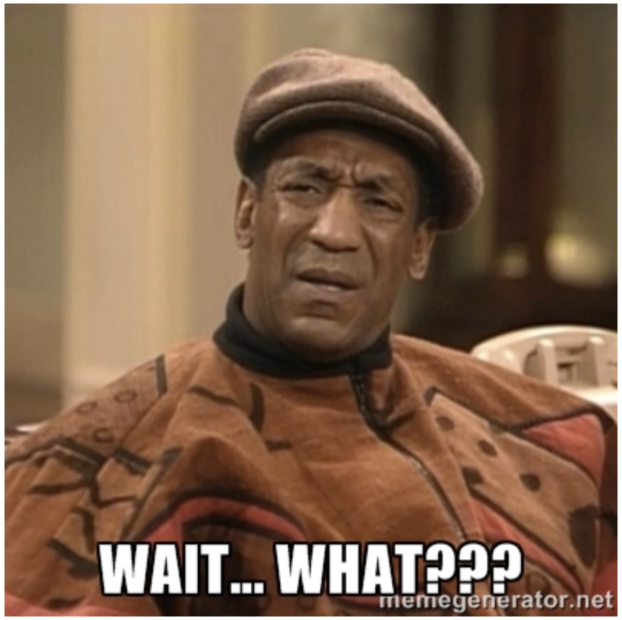
```
interface Loopback0
 ip address 10.0.1.1 …
 ip ospf 1 area 0
 ipv6 address FD00:DB8:1/128
 ipv6 ospf 1 area 0
```

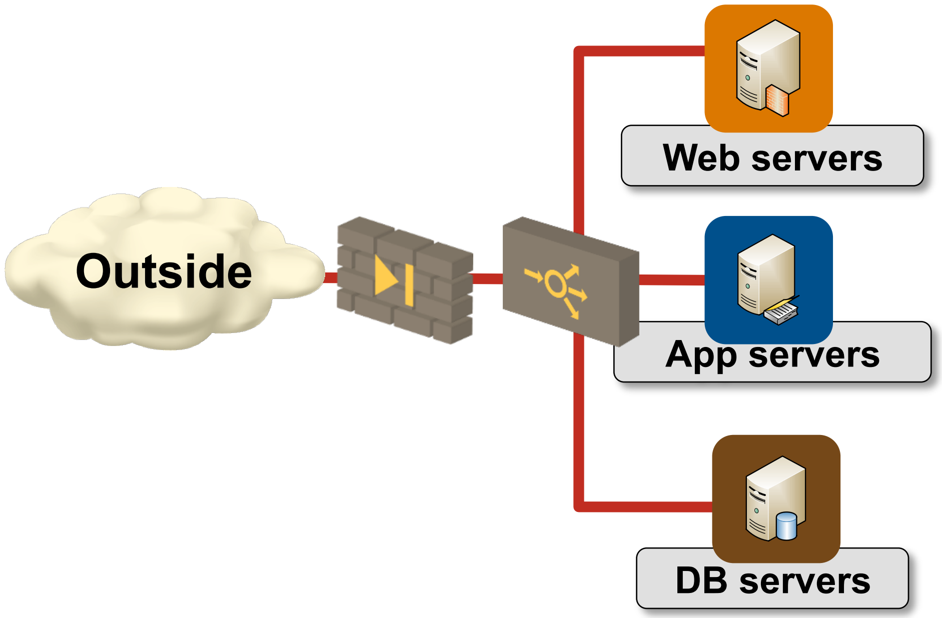
Failures Are Expensive: Real-Life Example

- Enable IPv6 in database segment ➔ OK
- Enable IPv6 in other segments ➔ OK
- Test connectivity ➔ OK

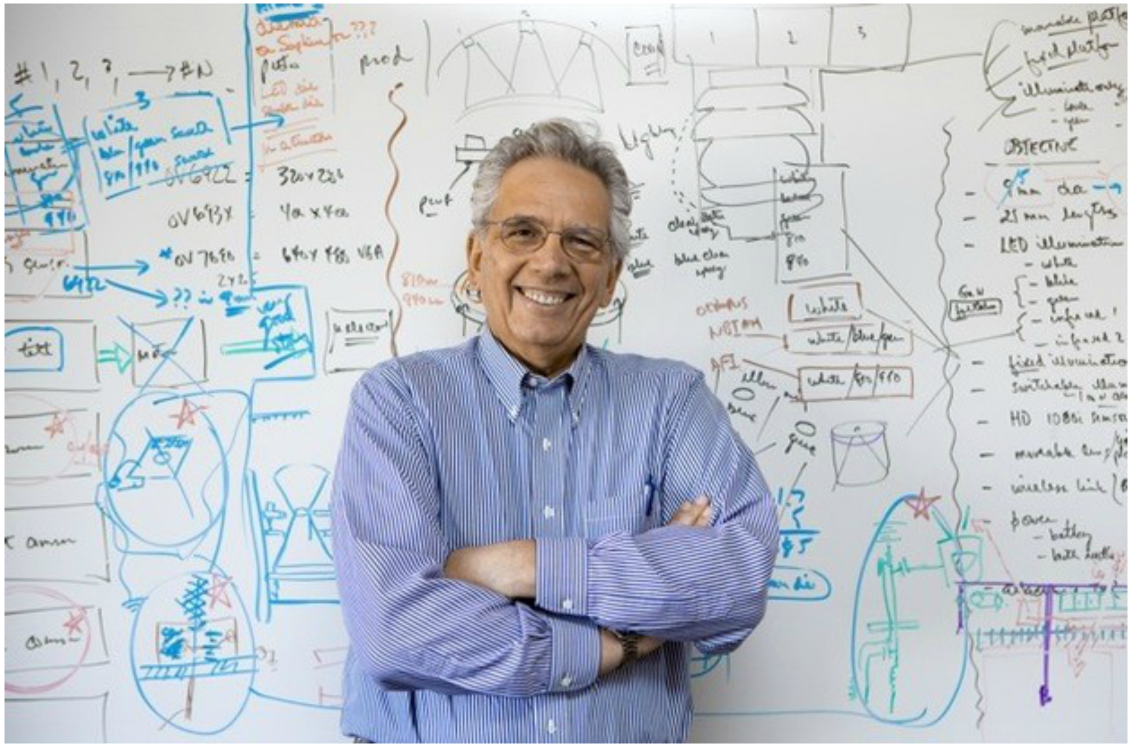
Weeks later…

- Add DNS server AAAA record ➔ **CRASH**





Every **Well-Defined** Repeatable Task Can Be Automated



Simplify

⇩

Standardize

⇩

Abstract

⇩

Automate

Collect

↓

Audit

↓

Cleanup

↓

Simplify

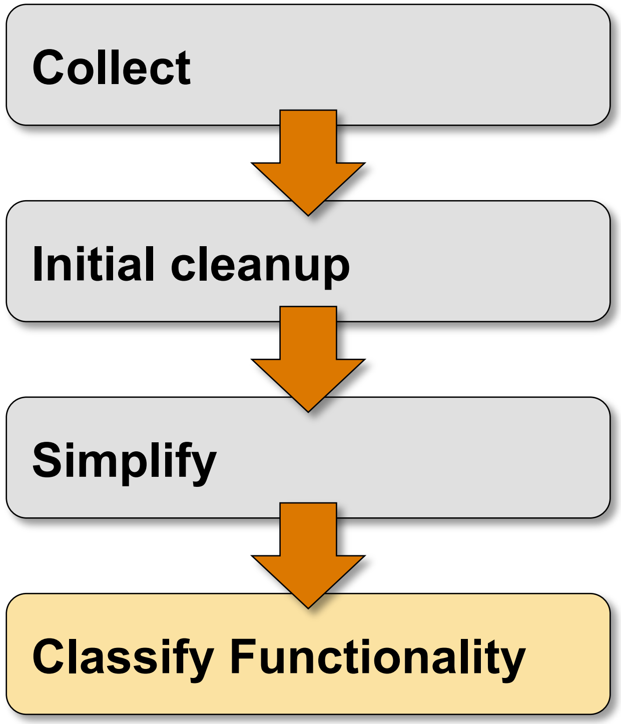
Preparation

Prepare for Migration: Functionality Classification

Identify parts of configuration that have to be migrated to IPv6

Potential classification outcomes:

- Functionality is not IP-dependent
- The functionality will remain on IPv4
- We need dual-stack functionality
- Functionality will move to IPv6



Prepare for Migration: Functionality Classification

Functionality is not IP-dependent

- Hostnames, usernames, passwords,

The functionality will remain on IPv4

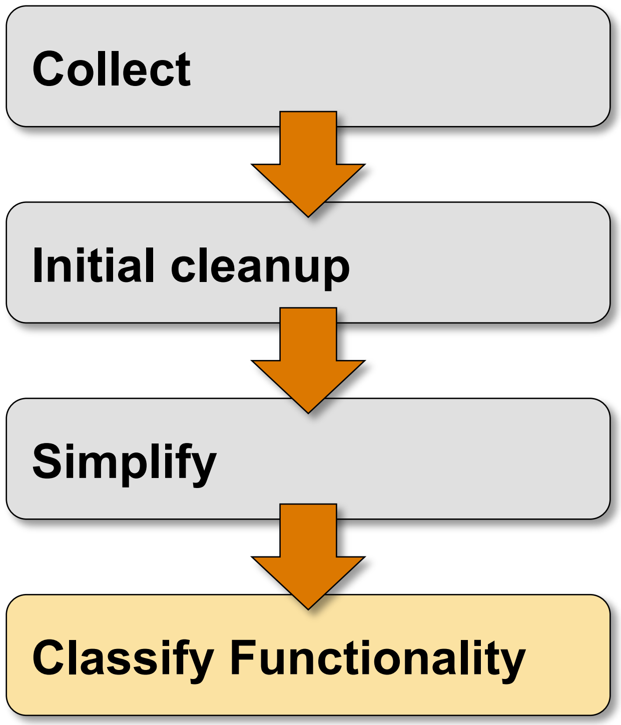
- Network management, MPLS control plane

We need dual-stack functionality

- Interface addresses
- Routing protocols
- Access lists and route maps

Functionality will move to IPv6

- Network management (?)



Classify Functionality: Examples

```
upgrade fpd auto
version 15.0
service timestamps debug datetime msec
!
hostname PE-A
!
boot-start-marker
boot-end-marker
!
logging buffered 4096
```

IP version agnostic

Ignore

Classify Functionality: Examples

```
interface Loopback0
 ip address 10.0.1.1 255.255.255.255
 ip ospf 1 area 0
!
logging host 172.16.1.12
!
snmp-server community cisco RO
snmp-server location Virtual
snmp-server host 172.16.1.12 cisco
!
track 100 interface Dialer3 ip routing
 delay down 10 up 10
```

Migrate to dual-stack

IPv4 only

Showstopper

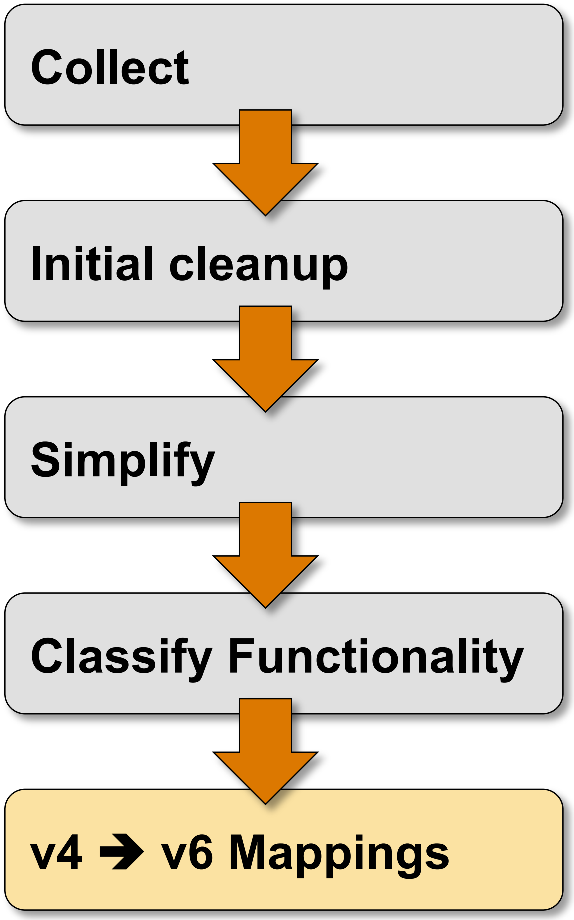
Prepare for Migration: v4 ➔ v6 Mappings

Add IPv6 equivalent of IPv4 configuration for every bit of dual-stack functionality

- Sounds simple
- Need well-defined v4 ➔ v6 mapping
- Where will you get it?

We need single source of (addressing) truth

```
interface Loopback0
 ip address 10.0.1.1 …
 ip ospf 1 area 0
 ipv6 address FEC0::CCCC:1/128
 ipv6 ospf 1 area 0
```

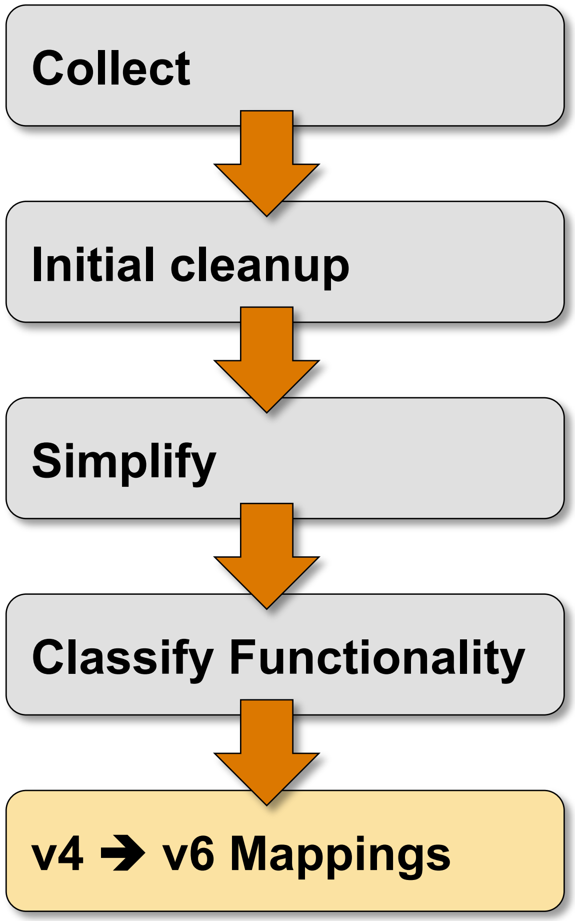


v4 ➔ v6 Mappings

Ideal use case: IPAM with hosts and subnets

Common: Excel spreadsheet

Worst case: no information



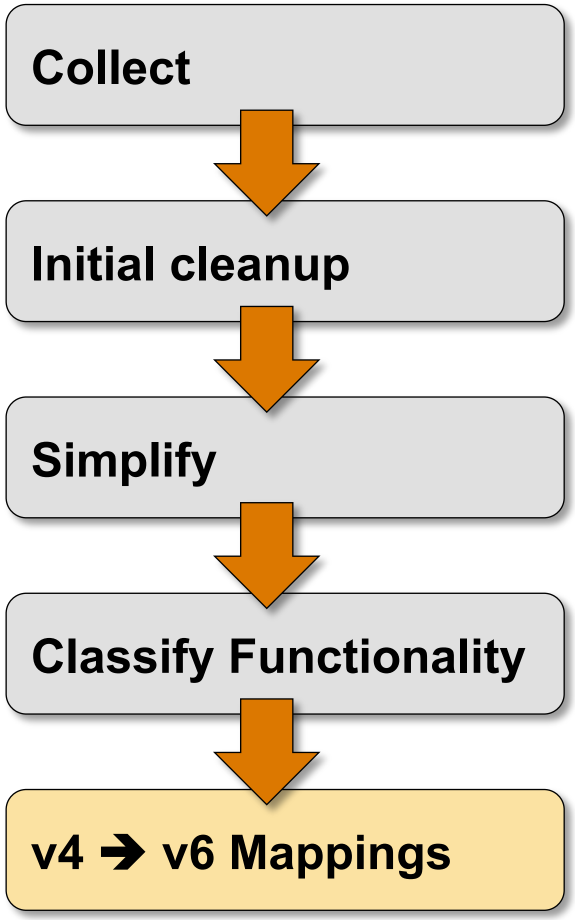
v4 ➔ v6 Mappings: Recovering from Worst Case

Assumptions:

- No IPAM (or reliable Excel)
- Device configurations are the only source of truth

Recovery process

- Analyze router configurations
- Scrape subnet information from interfaces
- Use simple algorithmic v4 ➔ v6 mapping to build IPv6 subnets and host addresses



Unfortunately we can't use DNS lookups

Getting the Job Done

Converting the Configurations: Screen Scraping

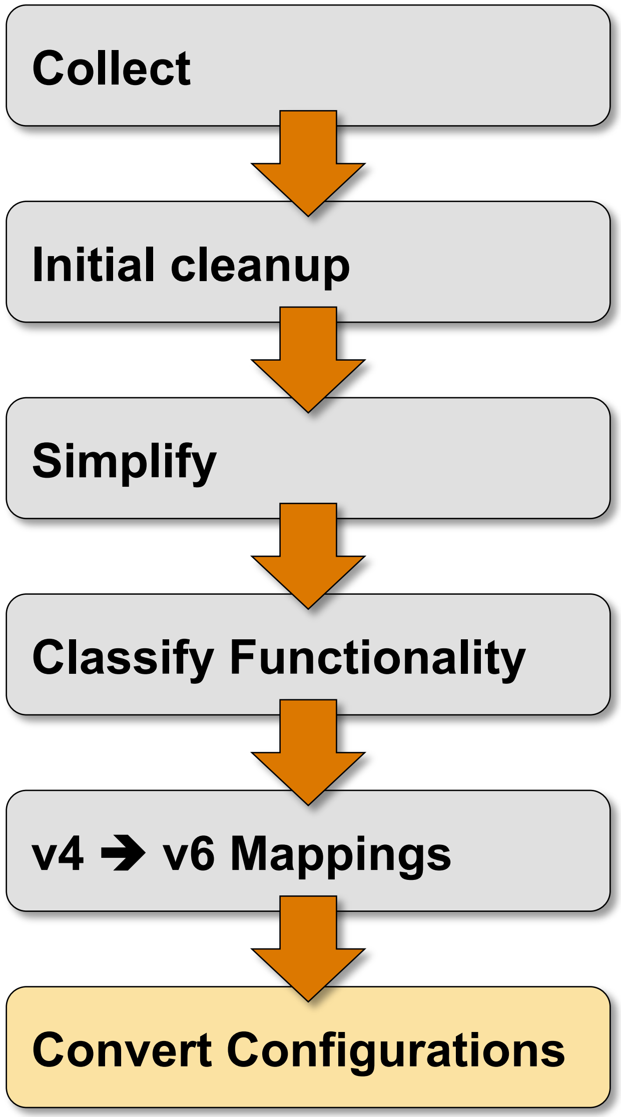
Perl is your friend ;)

Create a script that:

- Parses the configuration text file
- For every IPv4-specific command adds a corresponding (set of) IPv6 command(s)

Challenges: hierarchical objects

- Access lists
- Route maps
- Class maps and policy maps



Beyond Screen Scraping: Use XML or JSON

```
<interface>
  <Param>Loopback0</Param>
  <ConfigIf-Configuration>
    <ip>
      <address>
        <IPAddress>192.168.0.2</IPAddress>
        <IPSubnetMask>255.255.255.255</IPSubnetMask>
      </address>
    </ip>
    <ipv6>
      <address>
        <IPv6Prefix>FD00:DB0::1:1/128</IPv6Prefix>
      </address>
    </ipv6>
  </ConfigIf-Configuration>
</interface>
```

Converting the Configurations: Use JSON or XML

1. Download configuration in XML or JSON format
2. Traverse the XML or JSON object and create new object (or do an XSLT transformation)
3. Merge the new XML object with existing configuration (in script or in device)

Advantage: easier to program

Drawback: hard to check or track with version control tools

```
<interface>
  <Param>Loopback0</Param>
  <ConfigIf-Configuration>
    <ip>
      <address>
        <IPAddress>192.168.0.2</IPAddress>
        <IPSubnetMask>255.255.255.255</IPSubnetMask>
      </address>
    </ip>
    <ipv6>
      <address>
        <IPv6Prefix>FD00:DB0::1:1/128</IPv6Prefix>
      </address>
    </ipv6>
  </ConfigIf-Configuration>
</interface>
```

Real-Life Aspects

STOP WASTING YOUR TIME
ON THE LAST 10%

Real-Life Aspects

- Don’t expect to see a supported product – it will be a DIY job
- Pool the expertise – open-source and Github are your friends

Don’t try to be perfect:

- Identify the major challenges
  (addresses, subnets, ACLs, firewall rules)
- Automate as much as possible
- Lather, Rinse, Repeat…

One Does Not Simply
Deploy New Configurations

Real-Life Aspects

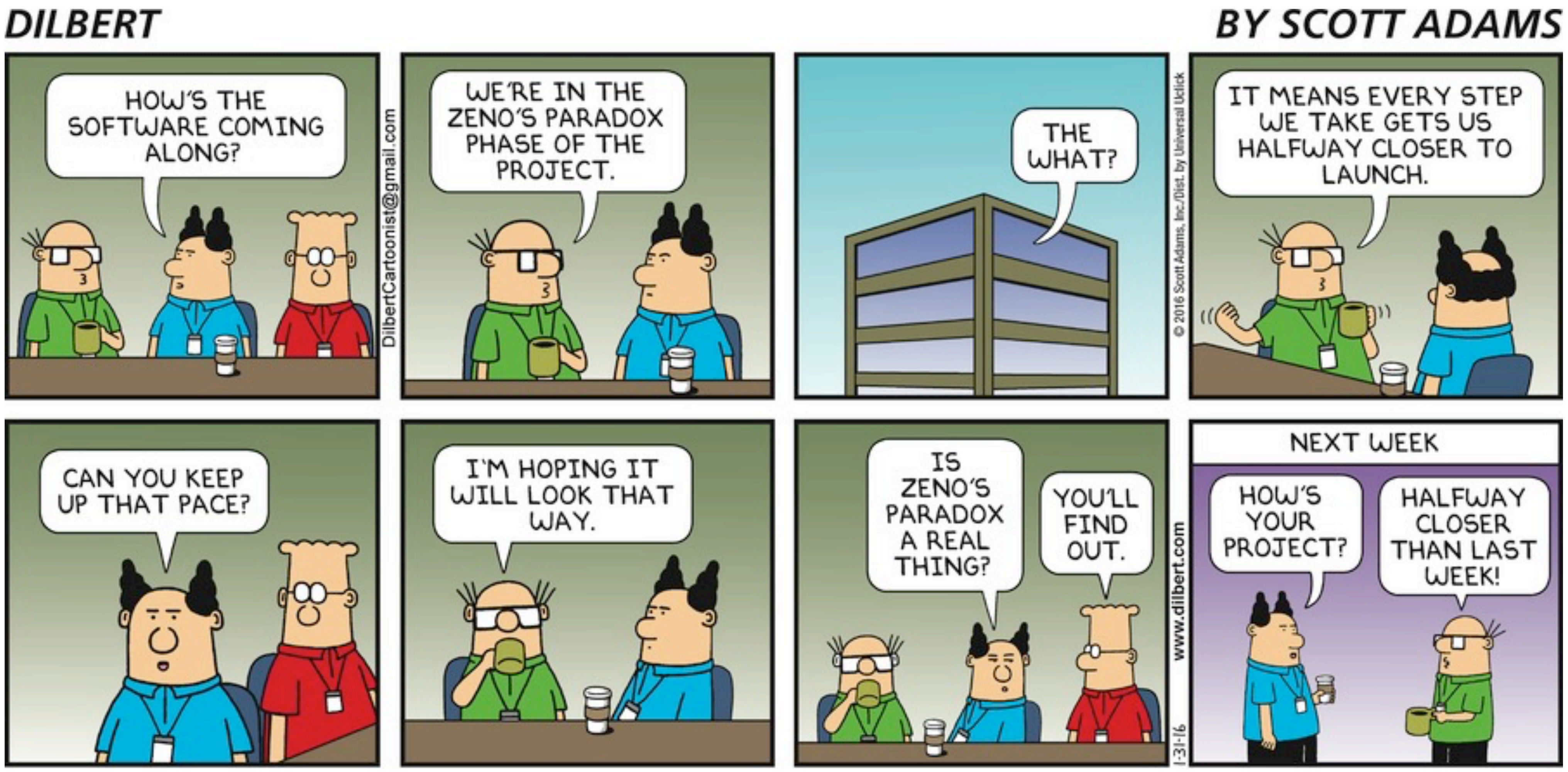
Let’s not waste time on change management, however…

- Source code management tools are your friends
- Use them to implement reviews and signoffs
- It’s better to have full configurations than snippets of new commands

Test, test, test…

- Write as many unit tests as possible for IPv4 world
- Convert these tests into IPv6 (using v4 ➔ v6 mapping)
- Execute IPv4 and IPv6 unit tests after every change

The Zeno's Paradox of Tool Development


DILBERT
BY SCOTT ADAMS
HOW'S THE SOFTWARE COMING ALONG?
WE'RE IN THE ZENO'S PARADOX PHASE OF THE PROJECT.
THE WHAT?
IT MEANS EVERY STEP WE TAKE GETS US HALFWAY CLOSER TO LAUNCH.
CAN YOU KEEP UP THAT PACE?
I'M HOPING IT WILL LOOK THAT WAY.
IS ZENO'S PARADOX A REAL THING?
YOU'LL FIND OUT.
NEXT WEEK
HOW'S YOUR PROJECT?
HALFWAY CLOSER THAN LAST WEEK!
DilbertCartoonist@gmail.com
© 2016 Scott Adams, Inc./Dist. by Universal Uclick
www.dilbert.com
1-31-16

Questions?

Send them to ip@ipSpace.net or @ioshints