# Distributed OpenSCAP Compliance Validation with MCollective

Trevor Vaughan - Onyx Point, Inc.

# Hi Everybody!

- Puppet Certified Professional
- Puppet Certified Developer
- Red Hat Certified Engineer
- Co-Founder of Onyx Point, Inc. (2009)
  - Puppet Labs Services Partner
  - Government Contracting
  - Automation, Data Flow, and Cloud Infrastructure Consulting
  - FOSS Supporters

# What We Will Cover

- Intro to SCAP
- Intro to MCollective
- The SCAP Security Guide
- Development Process
- Plugin Capabilities
- The Future
- Demo

# Introduction to SCAP

# What is SCAP

- Security Automation Content Protocol
  - NIST - 800-126
  - Language Definitions For
    - Configuration
    - Patch Checking
    - Vulnerability Checking
    - Technical Control Compliance
    - Security Measurement

# Relevant SCAP Languages

- XCCDF
  - Extensible Configuration Checklist Description Format
    - Provides mappings from Policy to Assessment
- OVAL
  - Open Vulnerability Assessment Language
    - Provides the actual checks against the system

# Why This is Important

- A recognized standard for Federal Systems
  - Often used for FISMA compliance checking
- Supported by most major vendors
- Ability to switch between approved tools *(or write your own!)*
- Everyone should support Open Standards!

SCAP

SECURITY GUIDE

# What is the SSG?

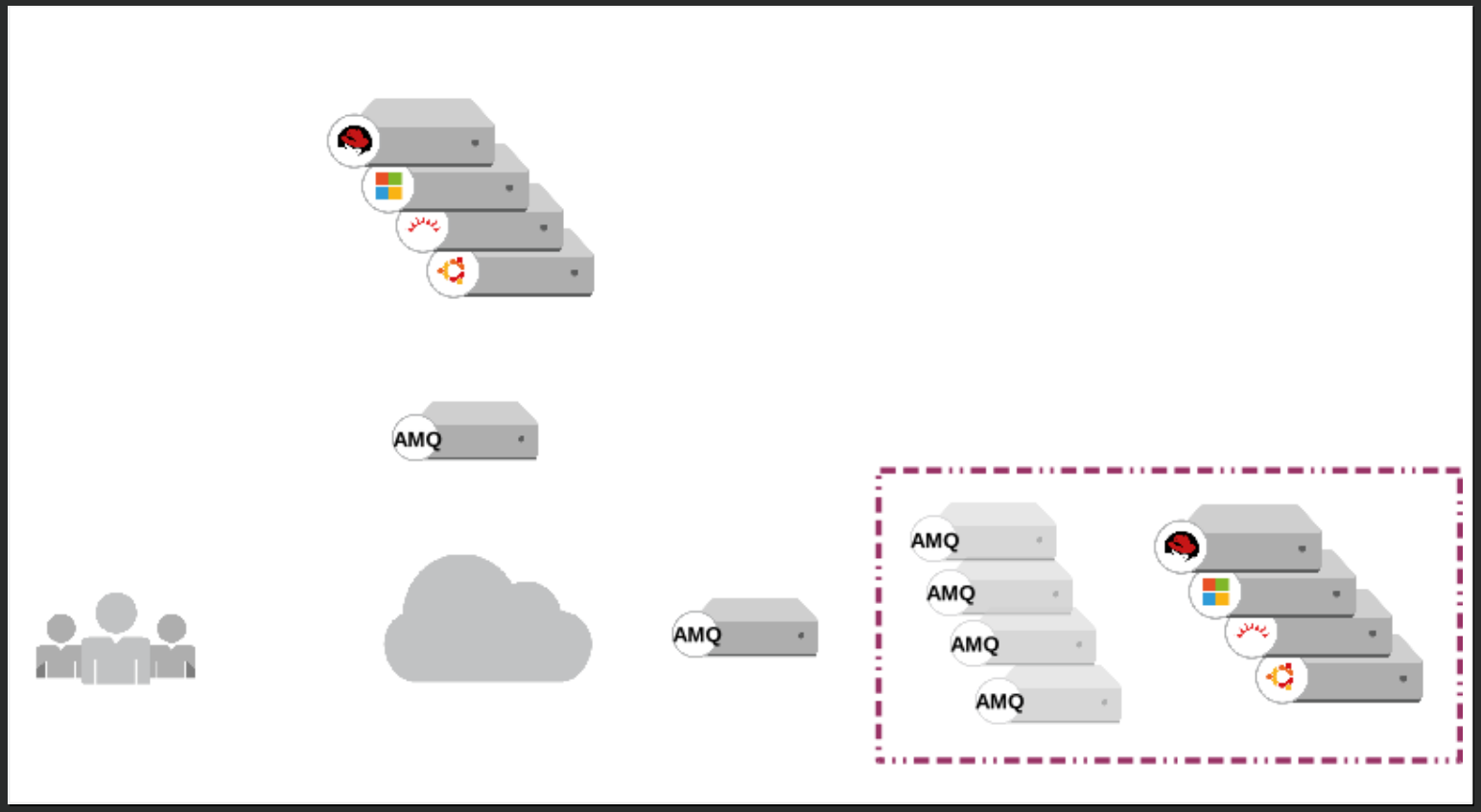
- Official SCAP baseline project for
  - Red Hat Enterprise Linux
  - Fedora Linux
  - Java
  - JBoss
  - OpenStack
- Upstream project for the DISA STIG
- Creators of USGCB Red Hat baseline content
- **GET INVOLVED!**
  - It's Open Source
  - Help Shape Rational Policy

# MCollective

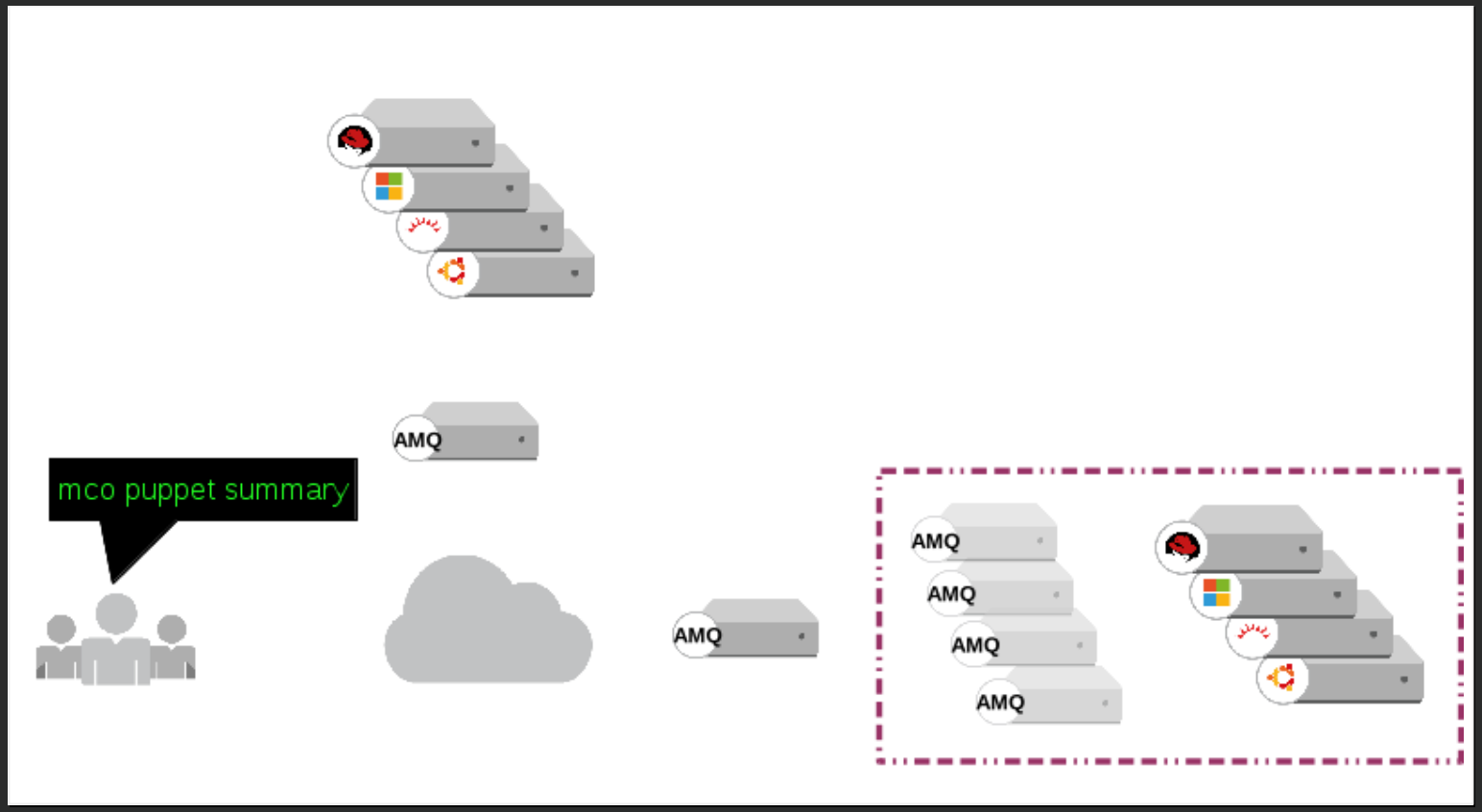
# What is MCollective?

- A Plugin-centric Command and Control Framework
- Designed to Work at Scale
  - Publish/Subscribe AMQP Middleware
- Security Friendly
  - Middleware Enables Few Port Connections
  - AMQP Provides Inbuilt Failover and Scaling
  - All Messages are Encrypted
    - Regardless of Transport
  - Plugin System
    - Enhanced Authentication/Authorization
    - Auditing and Restriction

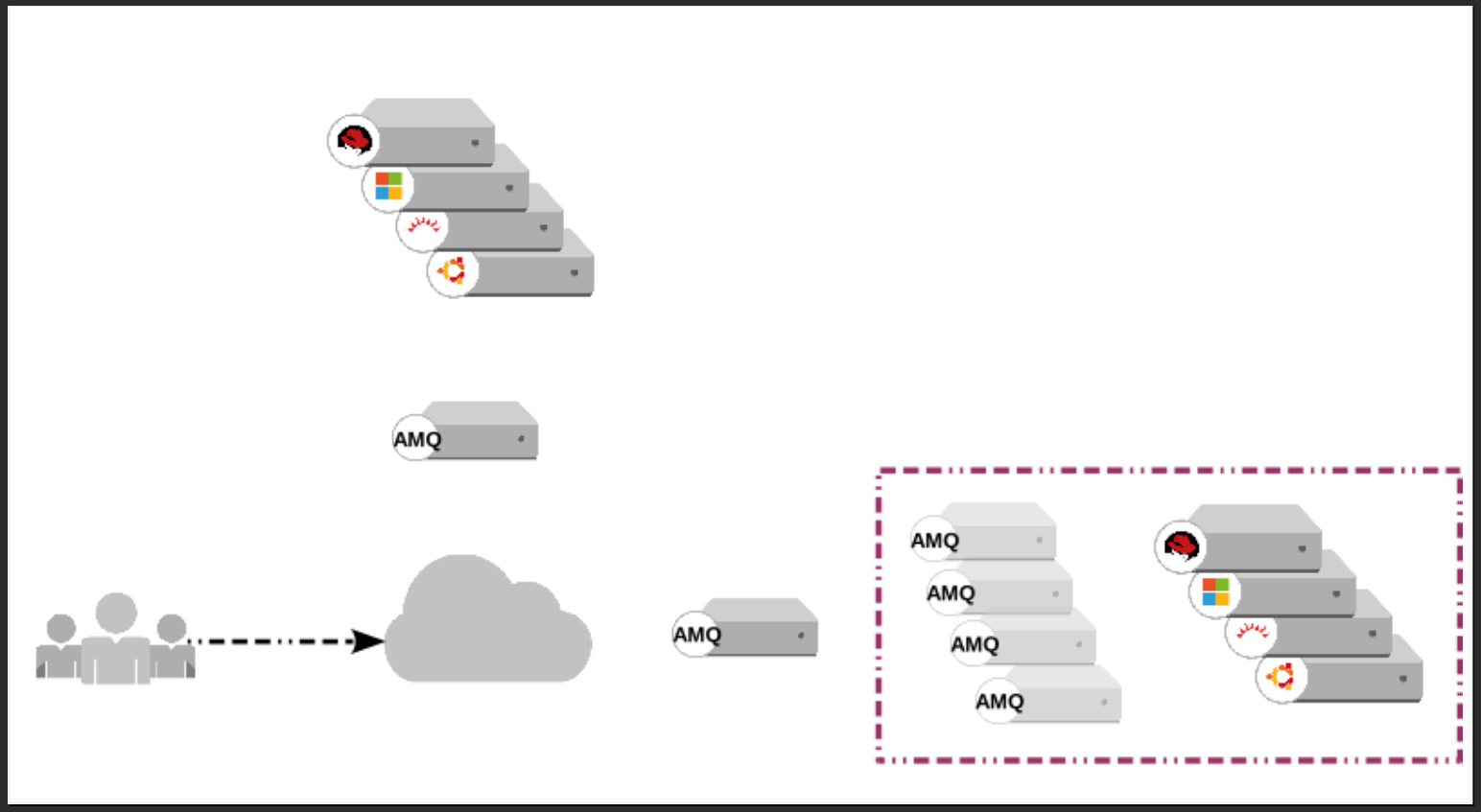
# MCollective Communication



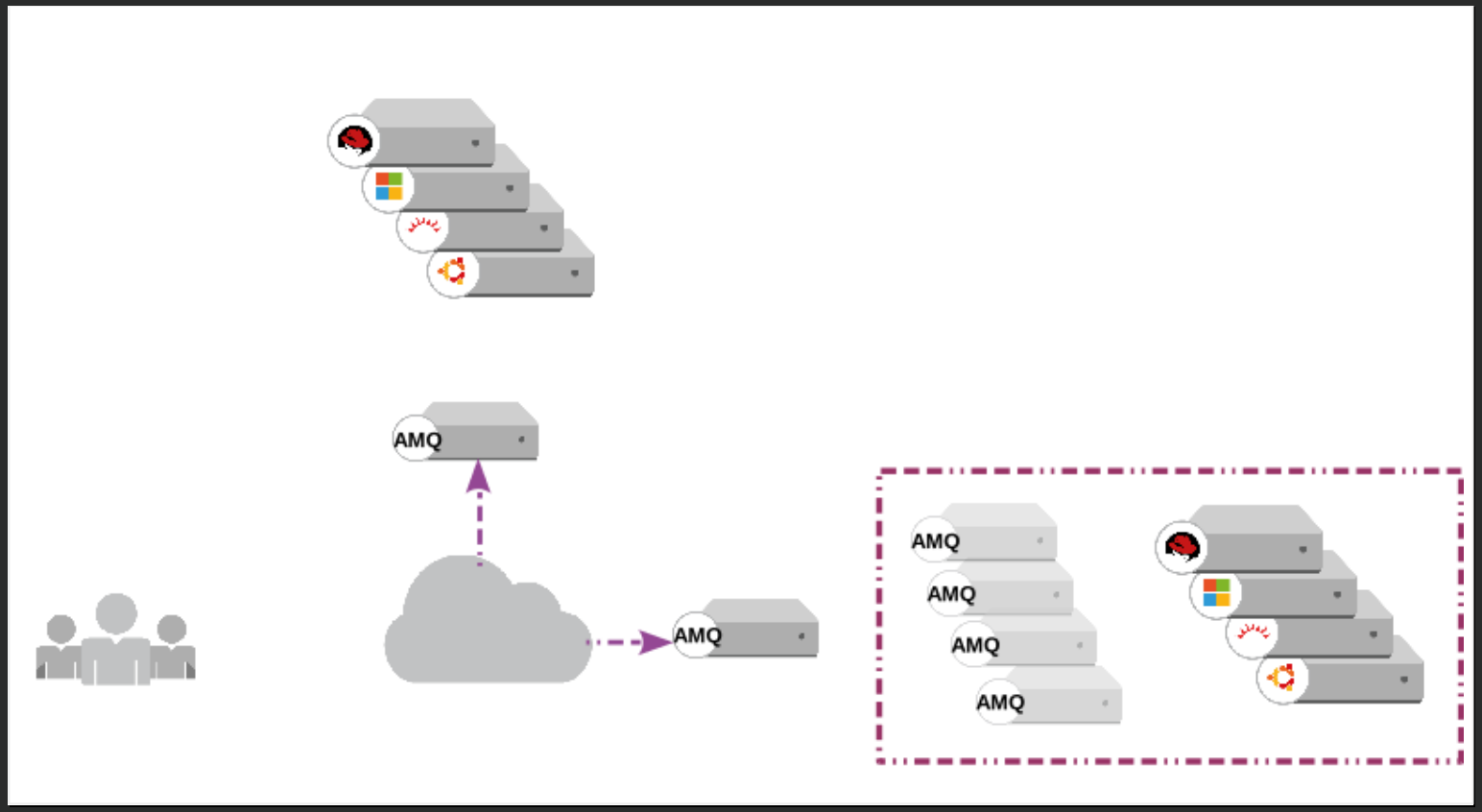
# MCollective Communication



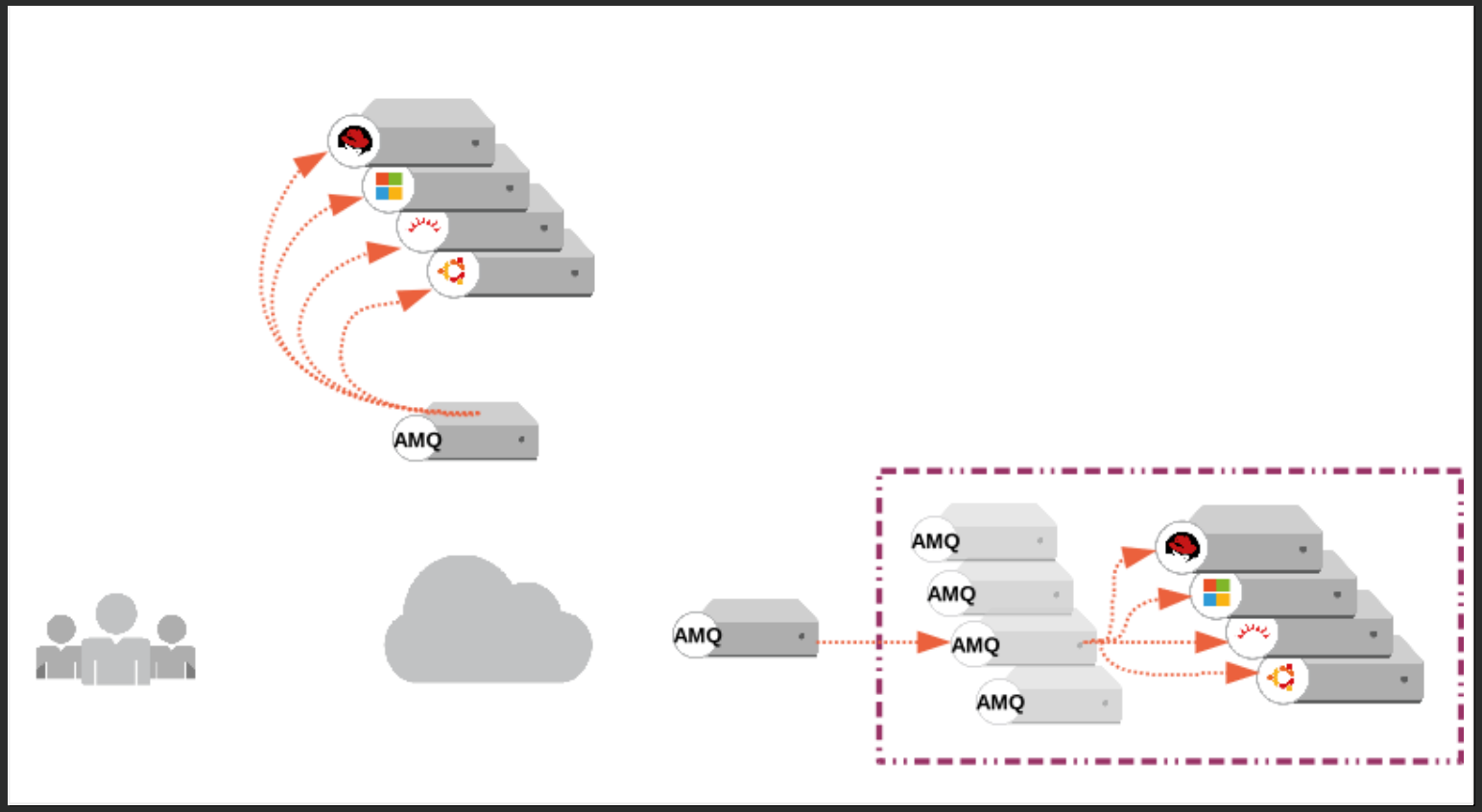
# MCollective Communication



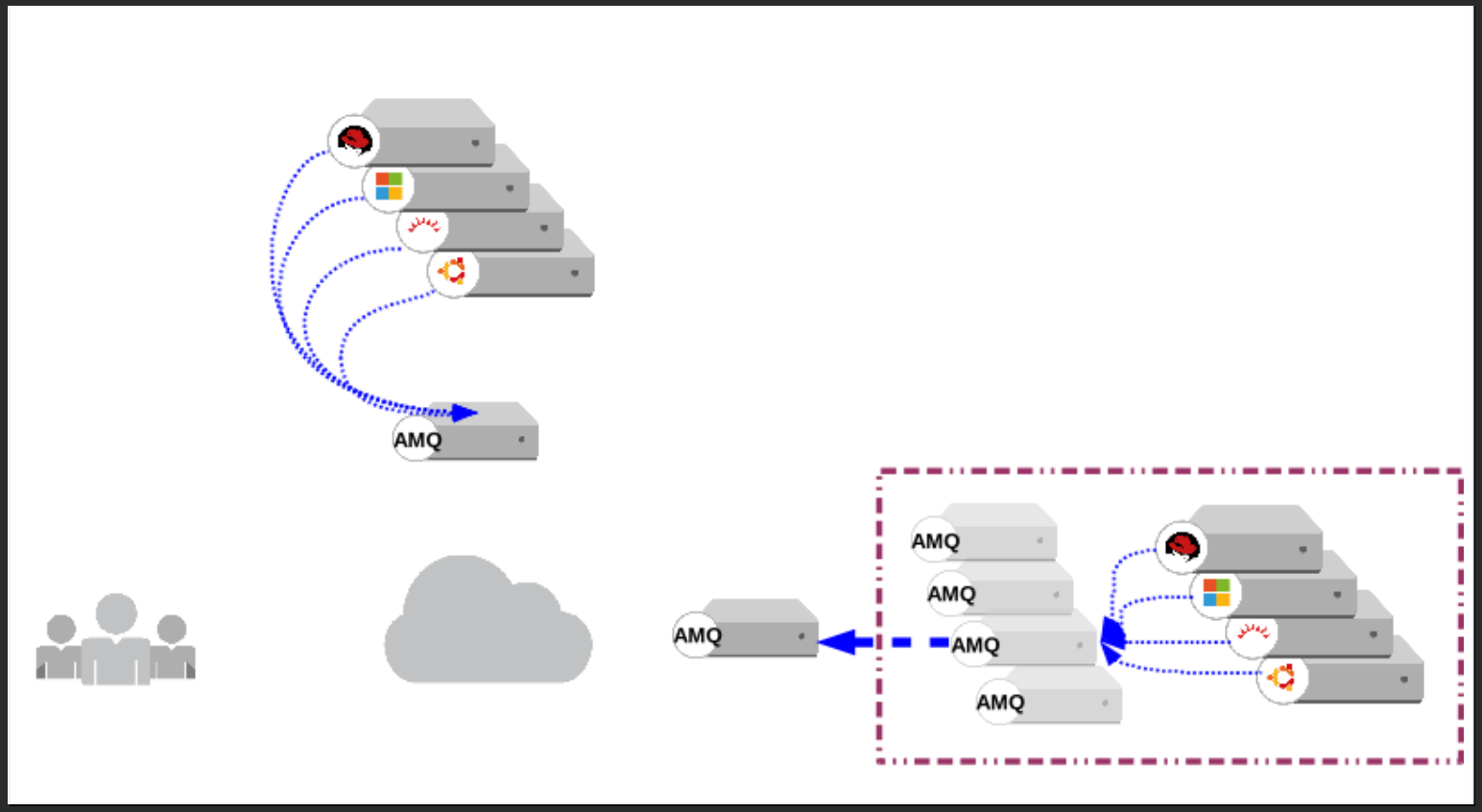
# MCollective Communication



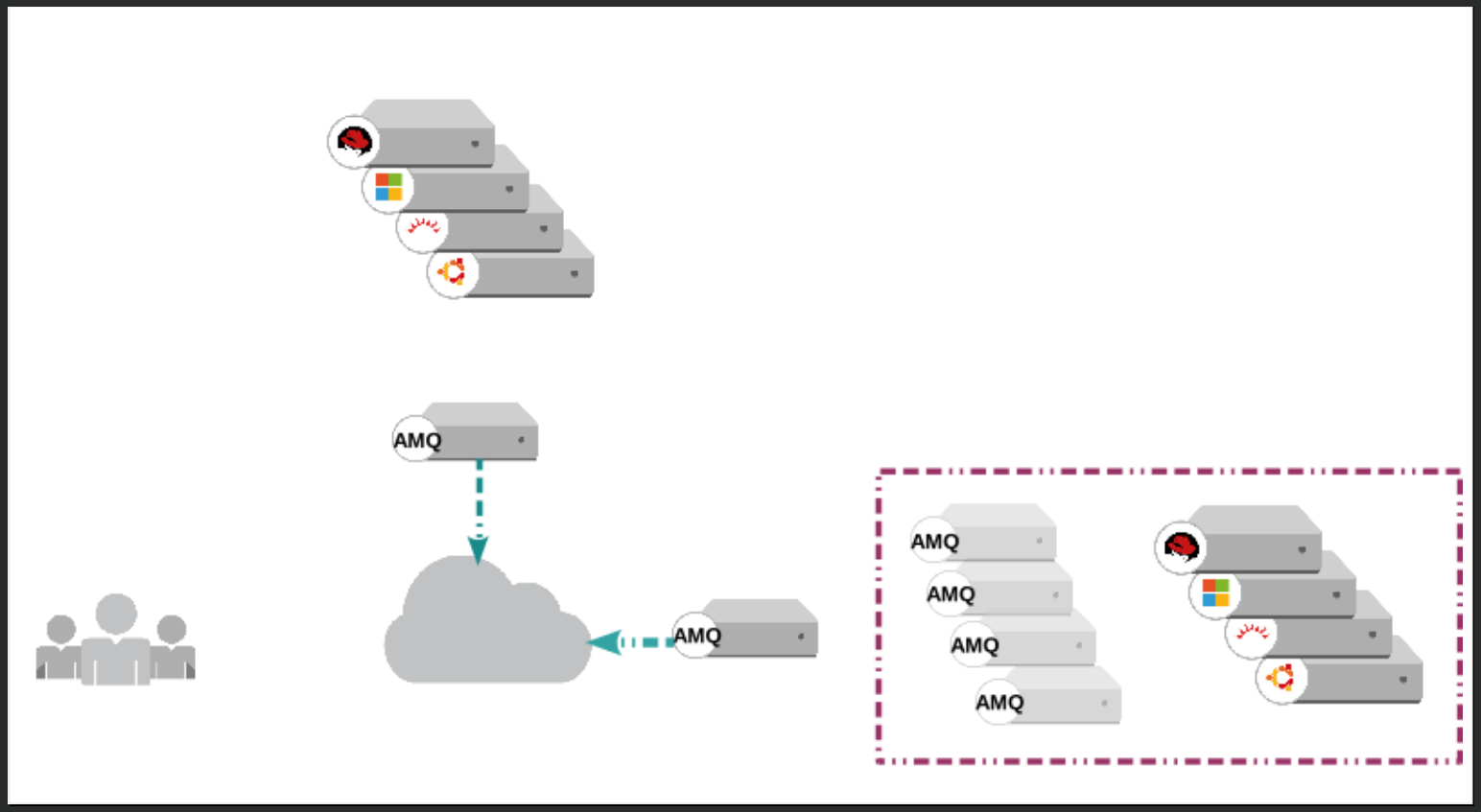
# MCollective Communication



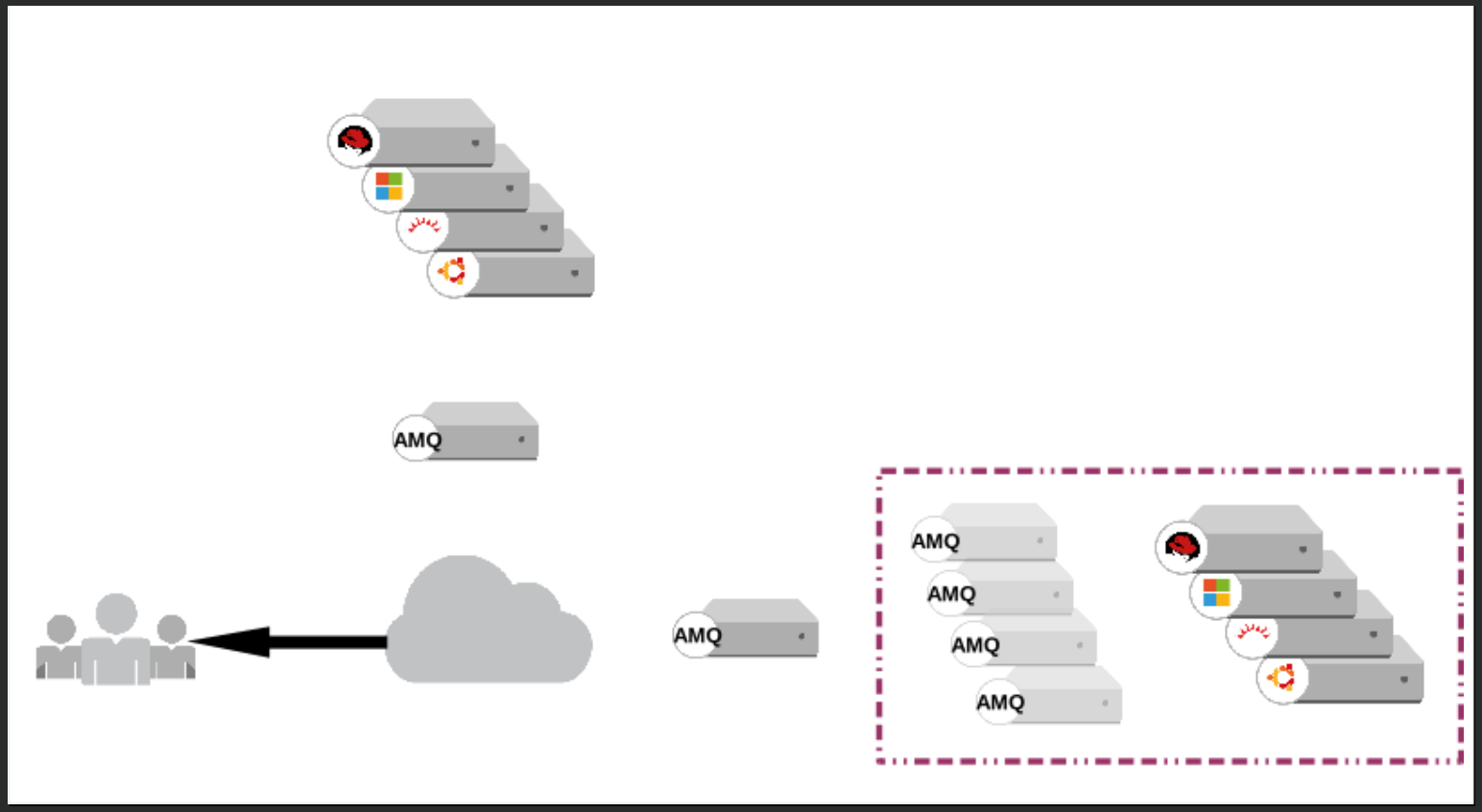
# MCollective Communication



# MCollective Communication



# MCollective Communication



# MCollective Communication

```
[rip@devco]% mco puppet summary
Summary statistics for 28 nodes:

      Total resources: ▅▂_____▂_▂_____ min: 340.00 max: 706.00
Out Of Sync resources: ▅_______▂_____ min: 0.00 max: 2.00
     Failed resources: ____________________ min: 0.00 max: 0.00
    Changed resources: ▅_______▂_____ min: 0.00 max: 2.00
Config Retrieval time: ▅▂▅▂_▂___▂▅▂▂ min: 3.88 max: 36.13
       Total run-time: ▅▂▅▂___▂____▂___ min: 7.08 max: 131.03
  Time since last run: ▅_________________ min: 15068.00 max: 141319.00
```

(R.I. Pienaar - Summary Sparklines)

# Plugin Development Process

# Writing the Agent: DDL

```
action 'scan', :description => 'Run an OpenSCAP scan.' do
  display :always

  # Required Parameters
  input :profile,
    :prompt       => 'Profile Name',
    :description  => 'A specific Profile to run.',
    :type         => :string,
    :validation   => '.*',
    :optional     => false,
    :maxlength    => 1024

  output :score,
    :description  => 'OpenSCAP Scan Score',
    :display_as   => 'Score',
    :default      => '0'

  summarize do
    aggregate summary(:score)
  end
```

# Writing the Agent: Capabilities

- Know what you need to run by hand **first**
- Remember: This part runs **on the server**

```
$ oscap xccdf eval --profile 'my-profile' --cpe cpe-dict.xml \
        --results /tmp/scan.xml os-xccdf.xml
```

- With SCAP, Pry and Nokigiri are your friends
  - Load the XML and dig for gold

# Writing the Agent: Functionality

1. Create your Scaffold
2. Add your *actions*
3. Rinse and Repeat

```
module MCollective
  module Agent
    class Oscap<RPC::Agent
      require 'mcollective/agent/oscap/util'
      include MCollective::Agent::Oscap::Util

      require 'mcollective/agent/oscap/profiles'
      include MCollective::Agent::Oscap::Profiles

      action 'profiles' do
        get_profiles(xccdf(request))
      end
    end
  end
end
```

# Writing the Application

- The User Interface to the System
- Independent Validation
- Receive and Process Results

```
def main
  rpcutil = rpcclient('oscap') # The name of your agent goes here
  printrpc rpcutil.send(configuration[:command],configuration)

  printrpcstats :summarize => true
end
```

# Testing!

- The Easy Way
  - Vagrant Up some boxes
  - Run 'mco oscap' a lot
- The Right Way
  - Rspec
  - Lots of examples with existing plugins
  - I'll get around to it one day ;-)

# Doing Horrible Things

- I have these great RHEL Profiles....
- But, can I run them on CentOS?
- Why Yes, now you can!
- Agents run Ruby and Ruby can manipulate data
  - Therefore...you can convert profiles on the fly!

No, this is **not** supported by the SSG team and my sincerest apologies to Shawn Wells

# Are You Awake?!

# Plugin Capabilities

# Operating System Support

- Currently Tested on RHEL7 and CentOS7
- Tested Against the SSG Profiles
- Other Systems **should** work

# Profile Discovery

- Need to know what profiles exist before scanning
- Mines the XCCDF file for a list of supported profiles
- Returns the list from all Nodes

```
$ mco oscap profiles
```

```
master
   OpenSCAP Profiles: ["rht-ccp"]

Finished processing 1 / 1 hosts in 172.97 ms
```

# OVAL Discovery

- Many times only a targeted scan is required
- No obvious list of what scan targets are availble
- Extracts the common name of plugins from the system

```
$ mco oscap oval_checks
```

```
master
   OVAL Checks: ["partition_for_tmp => oval:ssg:def:272",
                 "partition_for_var => oval:ssg:def:151",
                 "partition_for_var_log => oval:ssg:def:334",
# Lots more...
                 "snmpd_not_default_password => oval:ssg:def:164"]

Finished processing 1 / 1 hosts in 204.91 ms
```

# Performing a Full Scan

- Simplest scan form
- May take a **LONG** time

```
$ mco oscap scan -p rht-ccp -i ALL
```

```
master
   Scan Results:
          Score: 64.405869

Summary of Score:

   64.405869 = 1

Finished processing 1 / 1 hosts in 31973.00 ms
```

# Pfft...Scores are for the Weak

# Yep, That's a LOT of Data

```
$ mco oscap scan -p rht-ccp -i ALL -f
```

```
master
   Scan Results: {"partition_for_tmp"=>{
                    :severity=>"low",
                    :result=>"fail"
                  },
# 71 More Results...
                  "sshd_use_approved_ciphers"=>{
                    :severity=>"medium",
                    :result=>"fail"
                  }}
          Score: 64.405869

Summary of Score:

   64.405869 = 1

Finished processing 1 / 1 hosts in 10236.46 ms
```

# Something More Reasonable

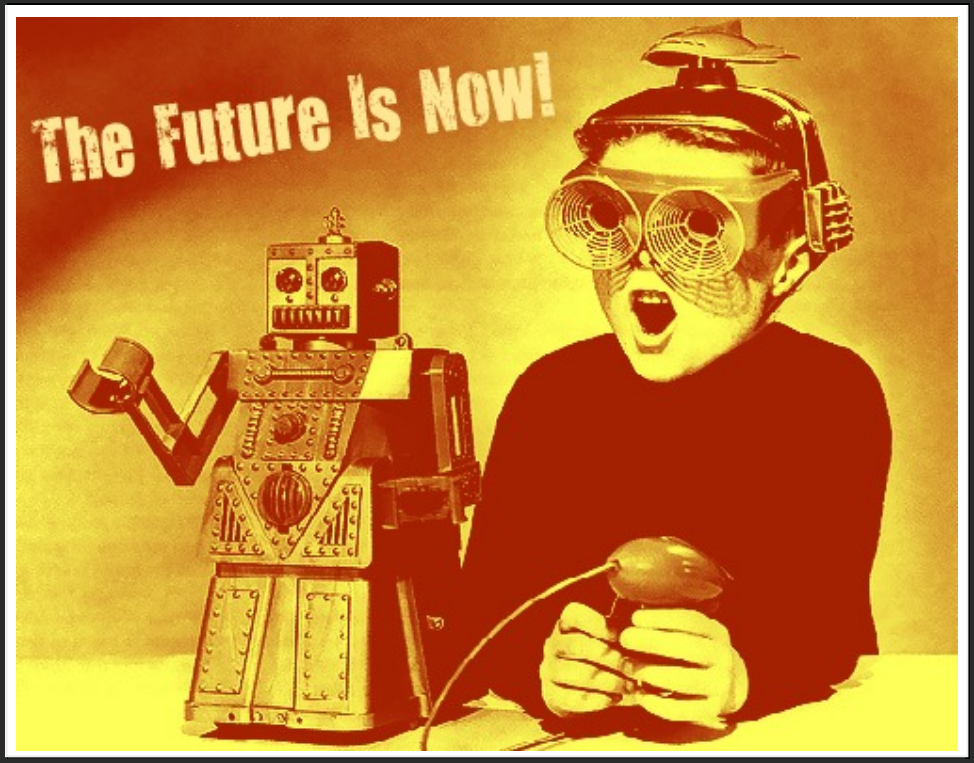
```
$ mco oscap scan -p rht-ccp -i package_telnet_removed
```

```
master
   Scan Results: Pass
          Score: 0

Summary of Score:

   0 = 1

Finished processing 1 / 1 hosts in 737.32 ms
```


The Future Is Now!

# **Future:** Automated Patch Scanning

- We can scan OVAL Content
- Vendors put out OVAL Patch Checks
- Security authorities should be able to scan systems as data is published

# **Future:** Profile Mangling

- We can already mangle Red Hat to CentOS
- Why not more?!
  - Only Target Scans of a Particular Level
    - Scan all nodes for High risk items
  - Disable Individual Checks
    - By regex or name
  - Disable Long Running Checks
  - Change Setting Thresholds on the Fly
    - Ex: Check that password length > 32
    - For targetd scans
    - Write a custom profile for large changes

# **Future:** Better Reporting

- Using the default reports for now
- Ideally would have rich summaries
- Complex analytics done elsewhere

# Future: LogStash Output

- For advanced reporting
- Send useful summary to MCO clients
- Send tagged data to LogStash
- Best of both worlds!

# Demonstration

peadmin@puppet:~

File Edit View Search Terminal Help

```
peadmin@puppet:~$ ./demo.sh
```

# Resources

- The Plugin
  - The MCollective Plugin for OpenSCAP
    - Please help make it better!
- Presentation Source
  - The source code for this presentation.
- Puppet Labs' MCollective Documentation
  - The official documentation on writing MCollective Agents and Applications.
- Learning MCollective Book
  - The definitive book on MCollective by Jo Rhett
- SCAP & STIG Workshop
  - Excellent SCAP Learning Material by Shawn Wells

# Presentation Information

This presentation was made possible by:

- Reveal.js by Hakim El Hattab

# Thanks for Coming!

**Questions?**