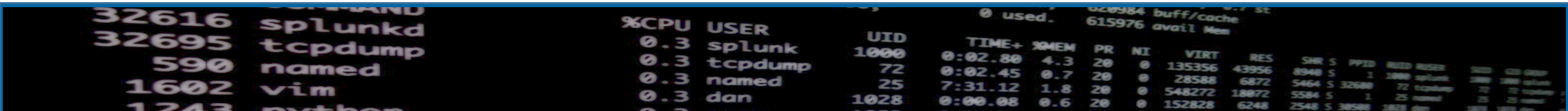
Managing Splunk with Puppet

Presented by Tom Kreiner

to Baltimore Splunk User Groups on

January 31, 2022

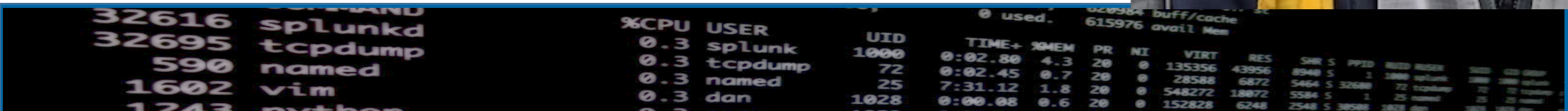


APLURA

Many Solutions, One Goal.

Who Am I?

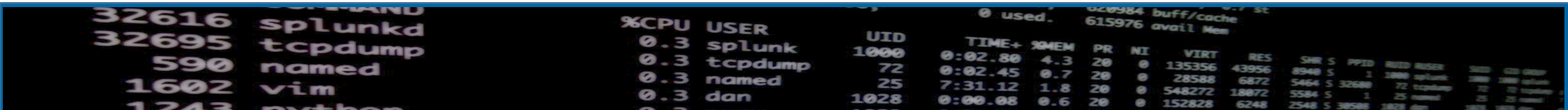
- Splunk professional since 2012
- Splunk Enterprise Certified Architect and Core Certified Consultant
- Joined Aplura in April 2021
- When not Splunking:
  - Husband, and father of two kids
  - Sailing
  - BBQ/grilling/smoking
  - Amateur radio



Many Solutions, One Goal.

Agenda

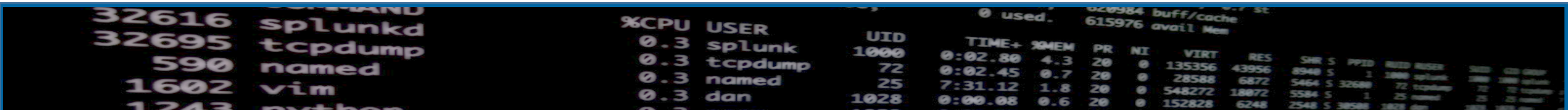
- Understand how Puppet can be used to install and configure Splunk.
- Learn benefits of this tool for both production and test environments.
- Discuss lessons learned of using Puppet, or any configuration management tool, with Splunk.



Many Solutions, One Goal.

WARNING – Lots of Information Ahead!!!

We are going to cover A LOT of information in the next hour. Use this session as a guide.



Many Solutions, One Goal.

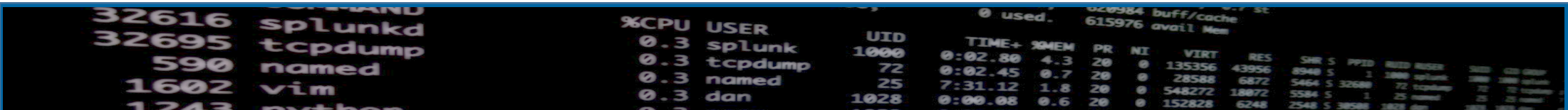
I Am Not a Puppet Master!!!

I have been working with Puppet for several months and I have learned A LOT!!!

I still have A LOT to learn!!!

Just like Splunk, there are lots of ways to solve the same problem.

Image from https://villains.fandom.com/wiki/Stromboli



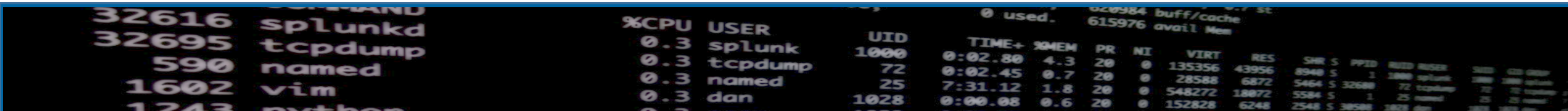
APLURA

Many Solutions, One Goal.

What is Puppet?

"Puppet is a tool that helps you manage and automate the configuration of servers."

-- Introduction to Puppet, puppet.com

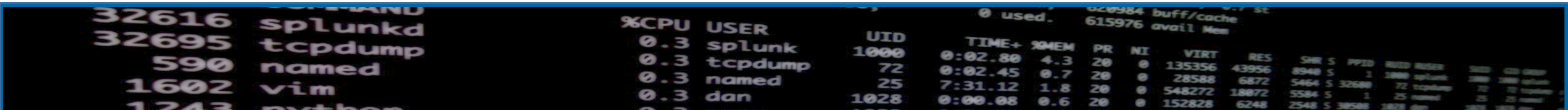


APLURA

Many Solutions, One Goal.

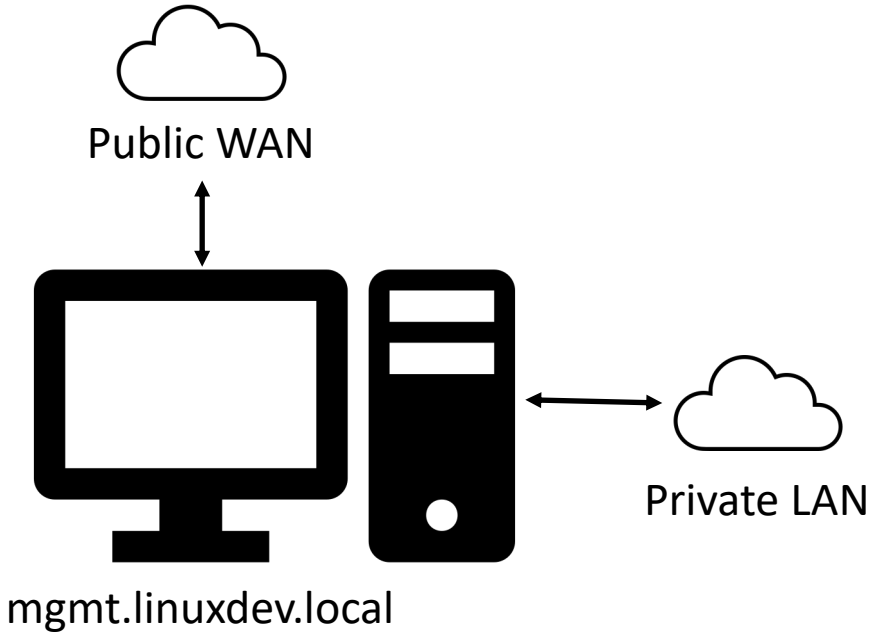
What is Benefit of Puppet with Splunk?

- Consistent install and configuration of Splunk.
- Time saving administration of servers.
- Easy provisioning of both production and test environments.
- Automation of upgrades.
- When managed in a repo, you gain history and change management of your configurations.

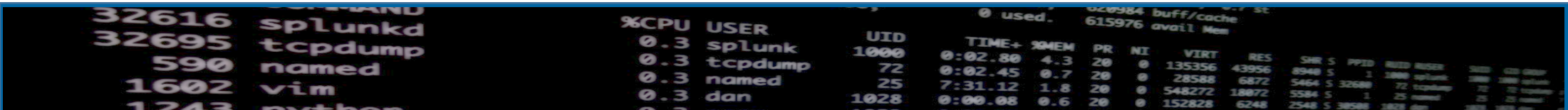


Many Solutions, One Goal.

My Environment – Management Server



- Router / Gateway
- DHCP
- DNS
- Package repository
- Puppet Server



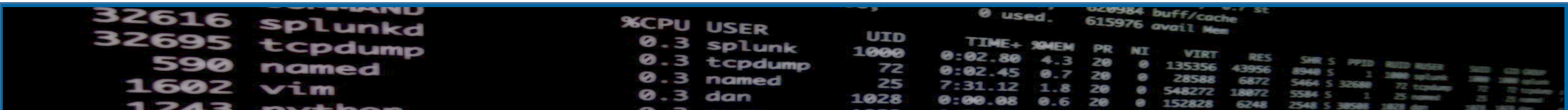
APLURA
Many Solutions, One Goal.

Important Concepts - Facter

Facter is a tool that installs with the Puppet agent that provides facts about a machine. Facts could include:

- OS information
- Disk configuration
- Network configuration
- Custom facts
- And so much more…

```
{
  architecture => "amd64",
  distro => {
    codename => "focal",
    description => "Ubuntu 20.04.3 LTS",
    id => "Ubuntu",
    release => {
      full => "20.04",
      major => "20.04"
    }
  },
  family => "Debian",
  hardware => "x86_64",
  name => "Ubuntu",
  release => {
    full => "20.04",
    major => "20.04"
  },
  selinux => {
    enabled => false
  }
}
```



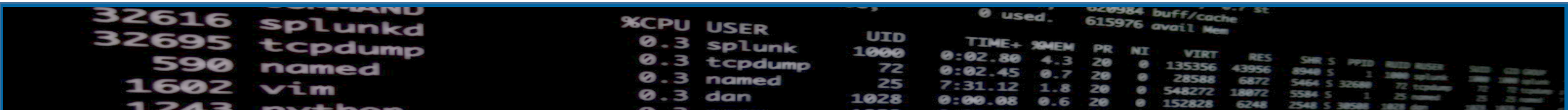
APLURA

Many Solutions, One Goal.

Important Concepts - Hiera

Hiera is a hierarchical configuration structure that is defined within the Puppet server to provide configuration settings. Facts from Facter are often used to define the structure. (NOTE: Think of Splunk app precedence with conf files.)

```
hierarchy:
  - name: "Encrypted data"
    lookup_key: eyaml_lookup_key
    paths:
      - "nodes/%{trusted.certname}.yaml"
      - "os/%{facts.os.name}.yaml"
      - "group/%{facts.hostname_info.group}.yaml"
      - common.yaml
```

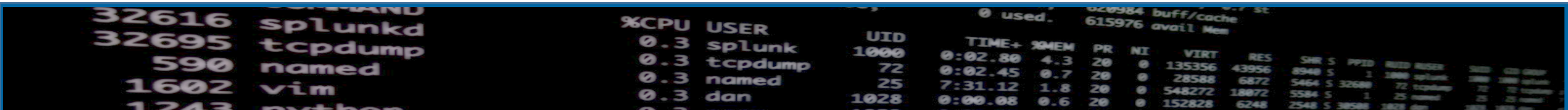


APLURA
Many Solutions, One Goal.

Important Concepts - Hiera

Here is a sample tree structure of Hiera data files that shows specific files created for my environment.

```
.
├── common.yaml
├── group
│   ├── srr.yaml
│   └── ugdemo.yaml
├── nodes
│   ├── mgmt.linuxdev.local.yaml
│   ├── splunk-clustermaster.linuxdev.local.yaml
│   ├── splunk-clustermaster-multi1.linuxdev.local.yaml
│   ├── splunk-clustermaster-multi2.linuxdev.local.yaml
│   ├── splunk-cmpeer1-multi.linuxdev.local.yaml
│   ├── splunk-cmpeer2-multi.linuxdev.local.yaml
│   ├── splunk-cmpeer3-multi.linuxdev.local.yaml
│   └── splunk-cmpeer4-multi.linuxdev.local.yaml
└── os
    └── Ubuntu.yaml
```



APLURA
Many Solutions, One Goal.

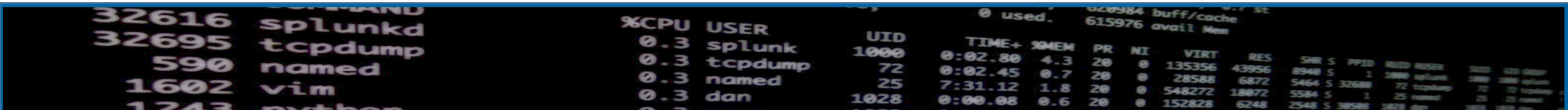
Important Concepts - Resource

A Puppet resource is a foundational object to define a configuration.

A resource is defined by a resource type and a name.

```
file { '/opt/splunk/etc/system/local/user-seed.conf':
  ensure  => file,
  owner   => 'splunk',
  group   => 'splunk',
  mode    => '0644',
  content => template('splunk/opt/splunk/etc/system/local/user-seed.conf.erb'),
}
```

```
user { 'suadmin':
  ensure     => present,
  groups     => $admin_groups,
  comment    => 'Test Sudo Admin User',
  home       => '/home/suadmin',
  managehome => true,
  shell      => '/bin/bash',
}
```

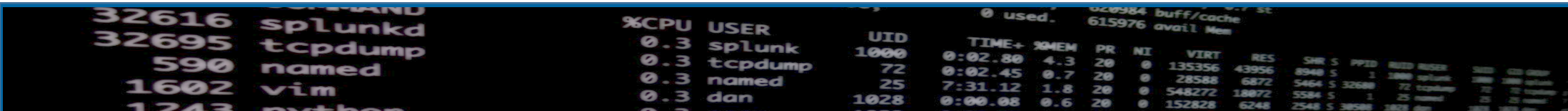


APLURA
Many Solutions, One Goal.

Important Concepts - Modules

A module is a combination of resources that provide some type of functionality.

```
class splunk inherits splunk::params {

  $config_vars = {
    version          => $splunk::params::version,
    splunk_user      => $splunk::params::splunk_user,
    splunk_password => $splunk::params::splunk_password,
  }

  package { 'splunk':
    ensure  => $splunk::params::version,
    require => [
      Apt::Source['tkreiner-app-repo'],
      Exec['apt_update'],
    ],
  }
```



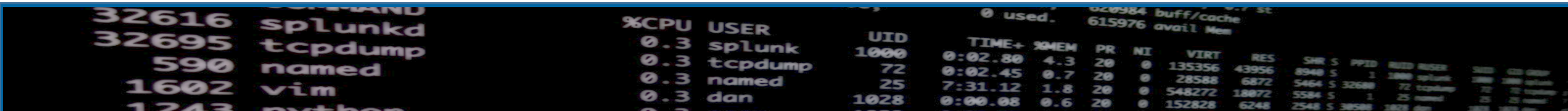
APLURA

Many Solutions, One Goal.

Important Concepts - Node

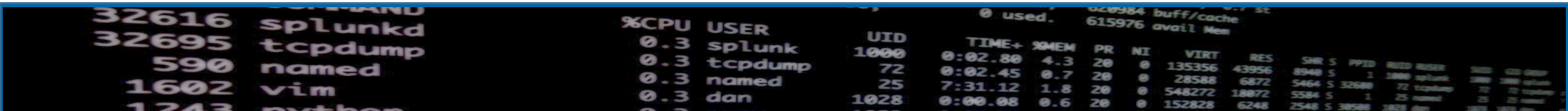
A node tells Puppet which configurations to apply to each of the agent nodes that connect.

```
# Splunk Servers
node /^splunk-standalone/ {
  include role::splunk_standalone
}

node /^splunk-clustermaster/ {
  include role::splunk_cm
}

node /^splunk-cmpeer/ {
  include role::splunk_cmpeer
}
```



APLURA
Many Solutions, One Goal.

Puppet Repo Demo



APLURA
Many Solutions, One Goal.

UG Demo Install

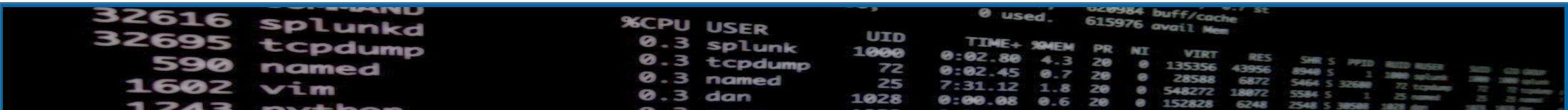
splunk-cmpeer1
Port 8001

splunk-clustermaster
Port 8000

splunk-cmpeer2
Port 8002

splunk-cmpeer3
Port 8003

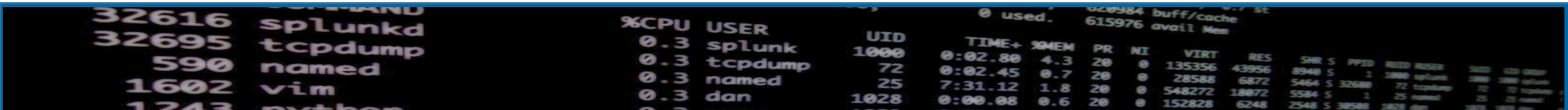
Using Vagrant, we are going to provision 4 servers to form a Splunk multi site cluster.



APLURA
Many Solutions, One Goal.

Vagrant Configuration

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/bionic64"
  config.vm.synced_folder "../provision/", "/provision"

  config.vm.define "splunk-clustermaster", primary: true do |box|
    box.vm.hostname = "splunk-clustermaster-ugdemo"
    box.vm.network "private_network", type: "dhcp", virtualbox__intnet: "intnet"
    box.vm.network "forwarded_port", guest: 8000, host: 8000
  end

  (1..3).each do |i|
    config.vm.define "splunk-cmpeer#{i}" do |box|
      box.vm.hostname = "splunk-cmpeer#{i}-ugdemo"
      box.vm.network "private_network", type: "dhcp", virtualbox__intnet: "intnet"
      box.vm.network "forwarded_port", guest: 8000, host: (8000+i)
      box.vm.provision "shell", inline: "/provision/set_attribute.sh splunk_site site#{(i % 3)+1}"
    end
  end

  config.vm.provision "shell", inline: "/provision/puppet_install.sh"
end
```

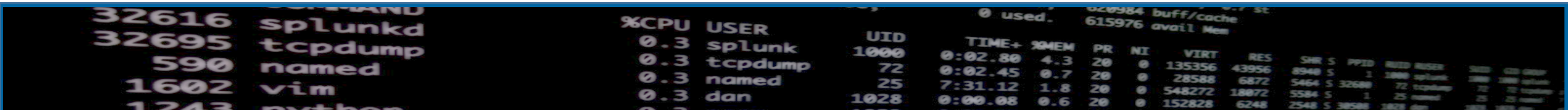


APLURA

Many Solutions, One Goal.

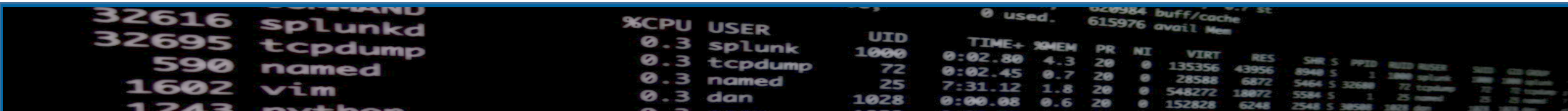
Hiera Configuration – data/group/ugdemo.yaml

```
splunk::params::version: '8.2.4'
splunk::params::cluster_label: 'splunk_ug_cluster'
splunk::params::multisite: true
splunk::params::site_replication_factor: 'origin:1, total:2'
splunk::params::site_search_factor: 'origin:1, total:2'
splunk::params::available_sites: 'site1,site2,site3'
splunk::params::manager_uri: 'https://splunk-clustermaster-ugdemo.linuxdev.local:8089'
splunk::params::forward_servers:
    - splunk-cmpeer1-ugdemo.linuxdev.local
    - splunk-cmpeer2-ugdemo.linuxdev.local
    - splunk-cmpeer3-ugdemo.linuxdev.local
```



APLURA
Many Solutions, One Goal.

Vagrant Provisioning Demo



APLURA

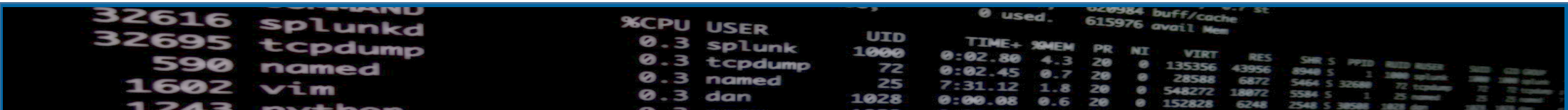
Many Solutions, One Goal.

Lesson #1 – Do Not Touch system/local

- Having Puppet manage configs in the system/local directory is a BAD idea.
- Splunk and Puppet will continuously fight.
- Example – Setting an indexer in detention causes a change in system/local/server.conf. Puppet detects the change and undoes it.

RESOLUTION

- Use application folders to manage configs. And…

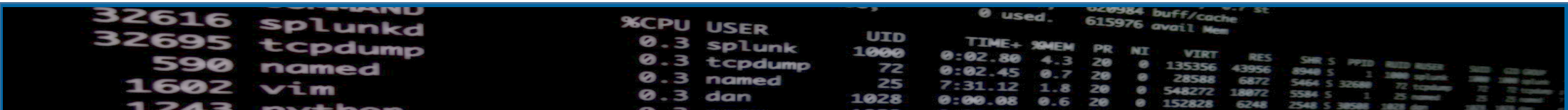


APLURA
Many Solutions, One Goal.

Lesson #2 – Add ”puppet” to app name

- Lesson #1 tells us to use applications to manage configs.
- When naming the application folder, use a naming convention that tells you that Puppet manages the app.

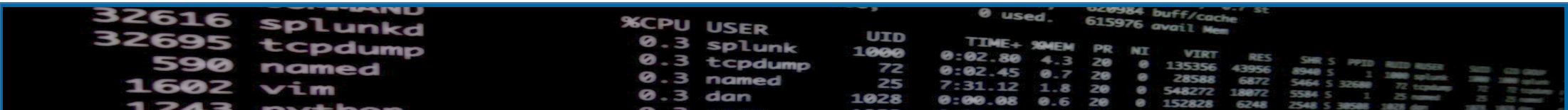
  Example: puppet_cm_config



APLURA
Many Solutions, One Goal.

Lesson #3 – Use Hashed Values

- When configuring sensitive information, (ssl_password, pass4SymmKey, etc.), make sure to have Puppet use hashed values.
- If you push a clear text password, Splunk will eventually convert it to a hashed value. However, Puppet will set it back when it detects a change in the file.
- This can cause Splunk to restart at times that you don't expect. Example: pushing a new index to a cluster will cause a restart.



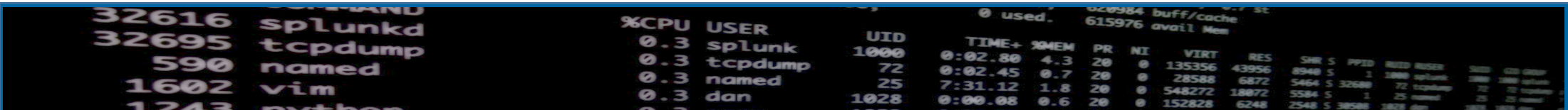
APLURA
Many Solutions, One Goal.

Lesson #4 – Beware of Service Starts

- Puppet can ensure that your Splunk service has started.
- This means that Puppet will ALWAYS make sure that Splunk is running.
- Beware of this, particularly if you are trying to decommission Splunk indexers. After an offline, Puppet may try to restart Splunk.

RESOLUTION

- You can use "puppet agent --disable" to disable the Puppet agent from running.



APLURA

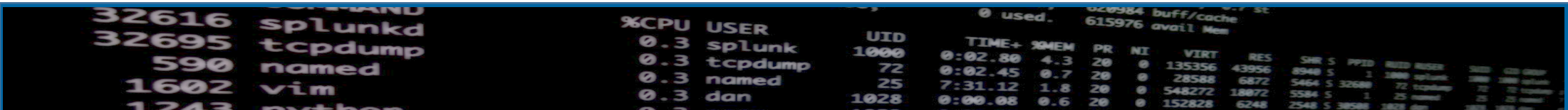
Many Solutions, One Goal.

Lesson #5 – Managing SHC Members

- Managing configurations in a search head cluster is tricky.
- Lesson #1 tells us to use apps to deploy configurations through apps. If you create an app in etc/apps in a SHC, the Deployer will remove it on the next bundle push.

RESOLUTION

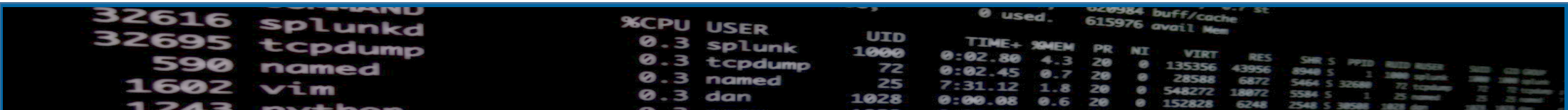
- Create the app in etc/apps and put all of your configs in the app's local directory.
- Create the same named app in Deployer. It will push all changes to the app's default directory.



APLURA
Many Solutions, One Goal.

Additional Resources

- Puppet Essential Training, LinkedIn Learning - https://www.linkedin.com/learning/puppet-essential-training
- Puppet Documentation - https://puppet.com/docs/puppet/7/puppet_index.html
- Puppet Forge - https://forge.puppet.com/



APLURA
Many Solutions, One Goal.