Live Response With Ansible

Presented By: Brian Olson

whoami - Brian Olson

Who

Manage Incident Response @ Verizon Media

Why I'm Here

Share an interesting solution

Situation

Recent Acquisition

Self-Reported Web Compromises

No knowledge of environment

Challenges

All public IP addresses

Colo's nationwide

No CI/CD process

Minimal staff/knowledge

Got Security?

- No Host IDS (HIDS)
- No Network IDS (NIDS)
- No Endpoint Detection & Response (EDR)
- Default Logs

My Toolbox

NSM

- Not on Corp Network
- Multiple Colo's Nationwide

EDR

- On-Prem solution
- Corp (internal) only

Ansible to the Rescue!

Adaptable

Not a security tool

Team Scope

- **Build & Operate Infrastructure**
- Detection
  - Host & network based
  - Big data analysis
  - Write detections
- Response
  - Digital Forensics
  - Incident Response

Remote Interaction	ssh | scp
Text Manipulation	cut | awk sed | grep
Loops	for | while

Stage 1: Bash-Foo

Bash-Foo Example

```
$ for x in $(cat hosts.txt); do echo $x; \
$ scp ./index.html ubuntu@$x:/var/www/html/; \
$ service apache2 restart; done
```

Bash Pros & Cons

Pros

- Quick & simple
- Familiar
- Iterate quickly
- Nothing special required

Cons

- Not easily reproducible
- No history/artifacts
  - Manual effort required
- Static
- **Serial execution**

Stage 2: DevOps Things

Chef	Ansible	Puppet	Salt
• Agent-based • Pull model • Ruby	• Agent-less • Push model • SSH & python	• Agent-Based • Pull model	• Agent-Based or Agent-less • Push or pull model

Why Ansible?

- SSH based
- Familiarity w/Ad-Hoc Mode
- Iterate quickly
- Fast adoption
- Abstract details
- Self-Documenting

Example Codeblocks

```
- name: Start apache service
  service:
    name: apache2
    state: started
    enabled: yes
  become: true
```

```
- name: Install LAMP packages
  package:
    name: "{{ item }}"
    update_cache: yes
    state: latest
  with_items:
    - apache2
    - mysql-server
    - php
    - php-mysql
  become: true
```

Ansible Enablers

- Required
  - Install python on local & remote hosts
  - Install ansible on mac/*nix (brew)
- Optional
  - Configure local ssh-agent
  - ansible.cfg
    - Inventory file
    - Remote user
    - Roles path

Ansible It!

Inventory File

- Static or Dynamic

Ad-Hoc Commands

- Do things quick'n'drirty

Modules

- Abstract the details

Playbooks

- Repeatability

Inventory File

[webserver]

54.32.59.20

3.80.126.**[23:57]**

webserver-**[01:33]**-denver.domain.com

[database]

34.12.87.123 # db1-denver

Ad-Hoc w/ OS Commands

Unix Name

ansible **all** -a “uname –a”

Restart Service

ansible **webservers** -a “service apache2 restart”

Process List

ansible **databases** -a “ps -ef”

Ad-Hoc w/ Ansible Modules

Package

ansible webservers **-m package** -a "name=apache2 state=present"

Service

ansible webservers **-m service** -a "name=apache2 state=restarted“

File

ansible webservers **-m file** -a “path=/web/main.php state=absent"

Ad-Hoc

```
~/git-repos/ansible-live-response  master ● ?  ansible all -a "uname -a"
54.167.10.226 | CHANGED | rc=0 >>
Linux ip-172-30-1-45 4.15.0-1032-aws #34-Ubuntu SMP Thu Jan 17 15:18:09 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
54.174.228.4 | CHANGED | rc=0 >>
Linux ip-172-30-1-60 4.15.0-1032-aws #34-Ubuntu SMP Thu Jan 17 15:18:09 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
~/git-repos/ansible-live-response  master ● ?
```

Triage Playbook

Get the Volatile Data

Running Processes

Netstat

Memory*

Download & Label Files

System & Web Logs

Bash History

Web Server Files

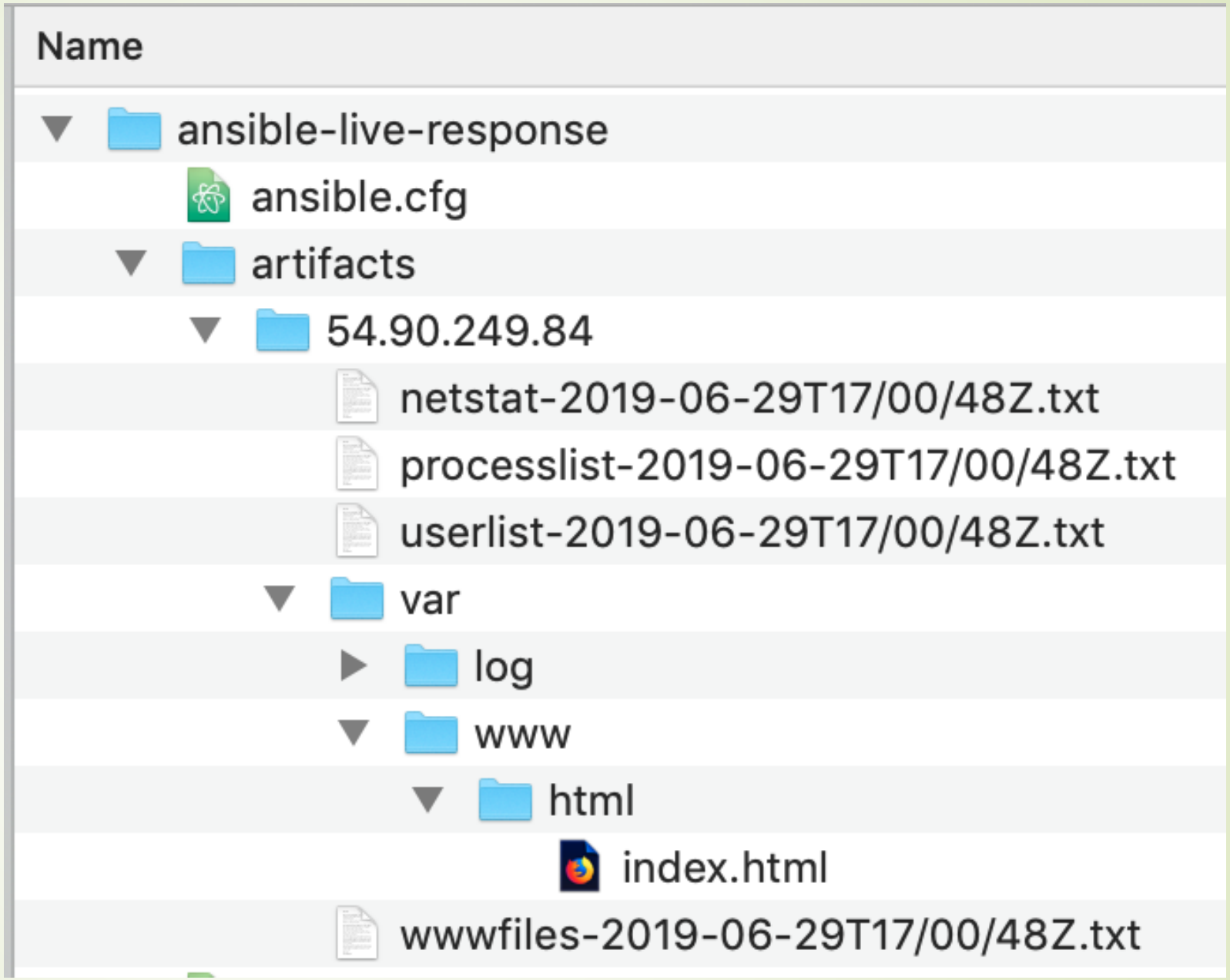
Possible Malware

```
---
# tasks file for DFIR-triage

- name: Make evidence collection directory ($pwd/artifacts)
  local_action:
    module: file
    path: artifacts/{{ inventory_hostname }}
    state: directory
    recurse: yes

### PROCESS DATA ###
- name: Get a list of all running processes from remote hosts
  shell: ps -ef
  register: ps_result

- name: Write remote process collection results to local artifacts
  local_action:
    module: copy
    content: "{{ ps_result.stdout_lines }}"
    dest: artifacts/{{ inventory_hostname }}/processlist-{{ansible_date_time.iso8601}}.txt
```

Triage - Playbook Sample

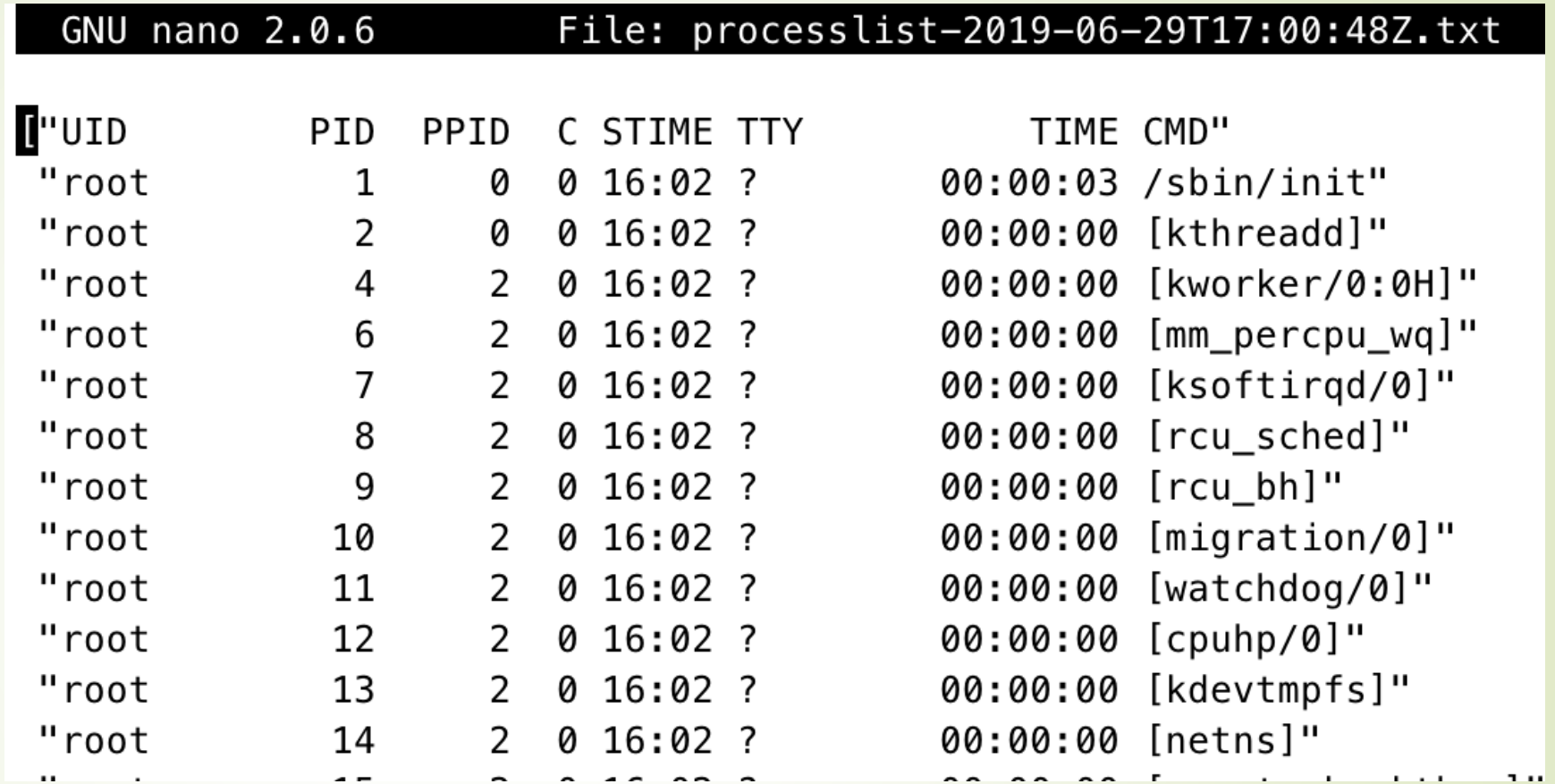
Artifact Collection

- Retain remote directory Structure
- Label & timestamp collected files



Artifacts

- Terminal-like data



Analysis

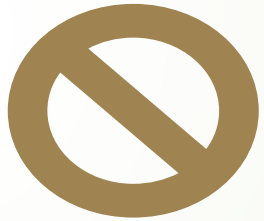
- Legit webdir files
- Stack analysis of webdir files
- Stack analysis of processes
- Unknown processes owned by apache
- Interesting network connections

Respond Playbook

Phase 1 – Stop the Bleeding

Patch hosts

Reconfigure Services



Phase 2 – Full Remediation

Remove malware

Remove unauthorized local users

Terminate suspicious processes & network connections

Respond – Playbook Sample

```
- name: Upgrade all  packages
  become: true
  package:
    name: "{{ item }}"
    update_cache: yes
    state: latest
  with_items:
    - default-jdk
    - apache2
    - mysql-server
    - php
    - php-mysql
  tags: phase_1
```

```
- name: Remove known malware
  file:
    state: absent
    path: /var/www/html/"{{ item }}"
  with_items:
    - malware.html
    - metasploit.html
    - poc.html
    - badness.html
  notify: restart_apache
  tags: phase_2
```

Triage - Demo

Summary

- Ansible is awesome for Live Response
- Scales reasonably well
- Modules abstract the details
- Collect all artifacts uniformly
- Playbooks standardize investigation & response
- Everything is easily customizable on the fly

Questions?

Brian Olson

E-Mail: brian@hurrikane.net

Twitter: @BrianOlsonSec

Github: https://github.com/brian-olson/ansible-live-response