HOW TO USE THIS DECK

This slide deck is meant to accompany the Ansible Security workshop, both sections.
Note that this deck is optional - the workshop content explains each and every Ansible idea in detail already.

HOW TO IMPROVE THIS DECK

The workshop is a collaborative effort. Help us to improve it! You can leave comments, and the BU will make sure to work on this. Tag for example Roland (Wolters) or Sean (Cavanaugh) to ensure that they pick it up.

Speaking about the BU: the fact that this deck is now owned by an organization and not individuals anymore hopefully ensures for the future that the deck stays up2date over time as the workshop develops.

Red Hat
Ansible
Automation

Ansible Security Automation Workshop

Introduction to Ansible Security Automation for System Administrators and Security Operators

Red Hat

Housekeeping

- Timing
- Breaks
- Takeaways

Red Hat

What you will learn

- Introduction to Ansible Automation
- How it works
- Understanding modules, tasks & playbooks
- How to use Ansible with various security tools
  - SIEM: QRadar
  - IDS: Snort
  - Firewall: Check Point NGFW

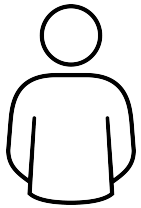
Introduction

Topics Covered:

- What Ansible Automation is
- What it can do

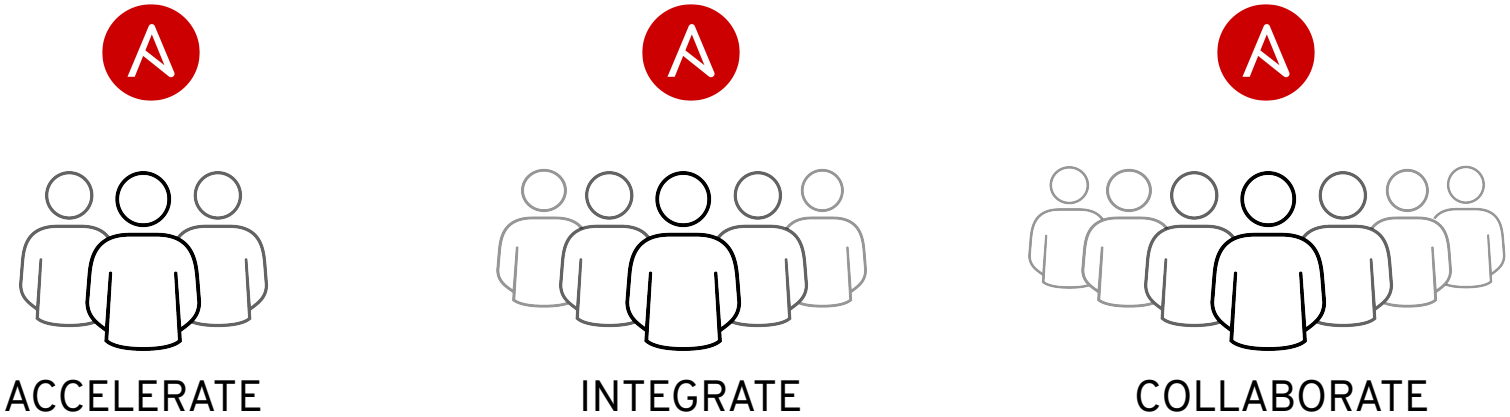
Red Hat
Ansible
Automation

Red Hat



Automation happens when one person meets a problem they never want to solve again

Red Hat



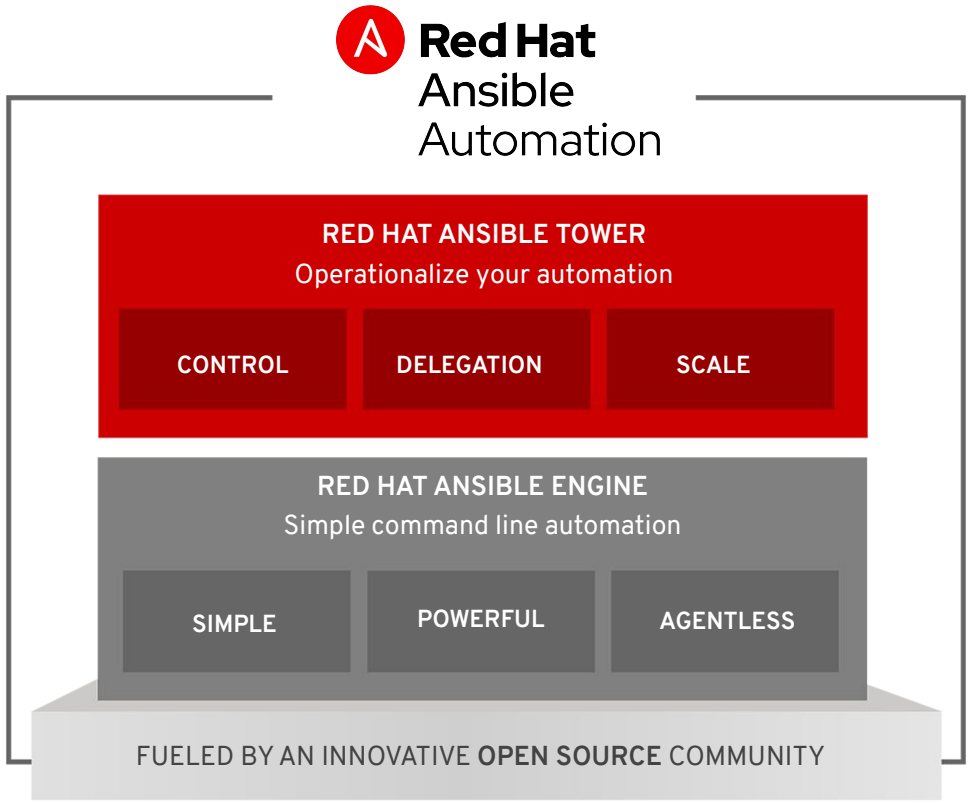
Red Hat

What is Ansible Automation?

Ansible Automation is the enterprise **framework** for automating across IT operations.

Ansible Engine runs Ansible Playbooks, the automation **language** that can perfectly describe an IT application infrastructure.

Ansible Tower allows you **scale** IT automation, manage complex deployments and speed productivity.



Red Hat

Why Ansible?

Simple

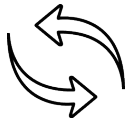
Human readable automation

No special coding skills needed

Tasks executed in order

Usable by every team

Get productive quickly



Powerful

App deployment

Configuration management

Workflow orchestration

Network automation

Orchestrate the app lifecycle

Agentless

Agentless architecture

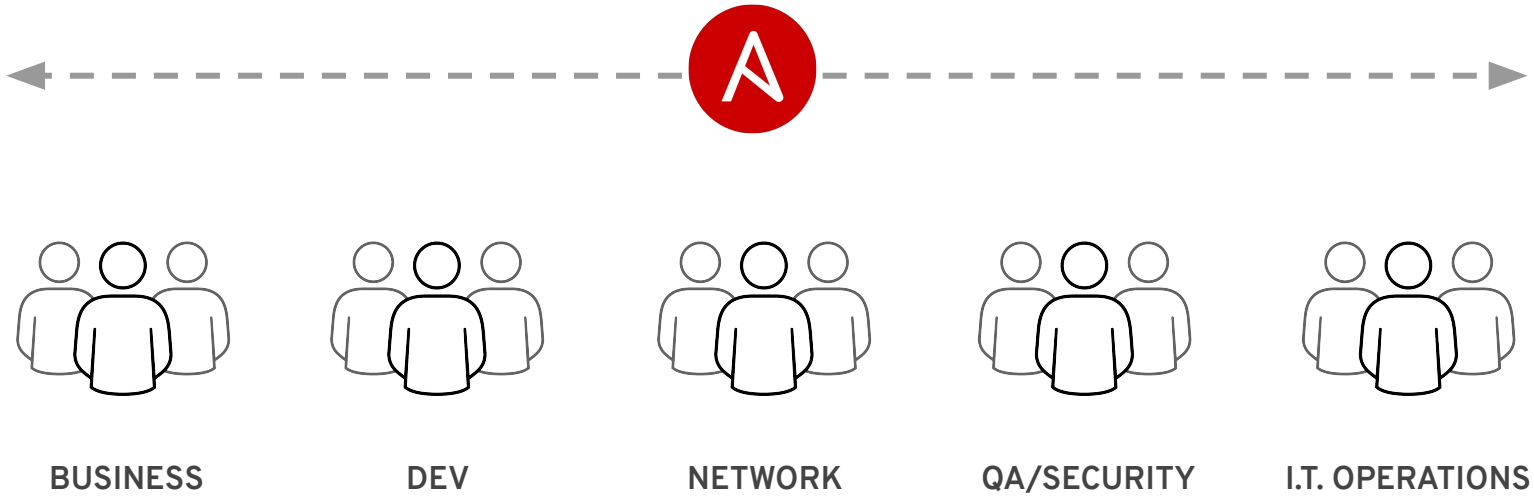
Uses OpenSSH & WinRM

No agents to exploit or update

Get started immediately

More efficient & more secure

Ansible Automation works across teams



Red Hat

What can I do using Ansible?

Automate the deployment and management of your entire IT footprint.

Do this...

- Orchestration
- Configuration Management
- Application Deployment
- Provisioning
- Continuous Delivery
- Security and Compliance

On these...

- Firewalls
- Load Balancers
- Applications
- Containers
- Clouds
- Servers
- Infrastructure
- Storage
- Network Devices
- And more...

Red Hat

Ansible automates technologies you use

Time to automate is measured in minutes

Cloud

AWS
Azure
Digital Ocean
Google
OpenStack
Rackspace
+more

Operating Systems

Rhel And Linux
Unix
Windows
+more

Virt & Container

Docker
VMware
RHV
OpenStack
OpenShift
+more

Storage

Netapp
Red Hat Storage
Infinidat
+more

Windows

ACLs
Files
Packages
IIS
Regedits
Shares
Services
Configs
Users
Domains
+more

Network

Arista
A10
Cumulus
Bigswitch
Cisco
Cumulus
Dell
F5
Juniper
Palo Alto
OpenSwitch
+more

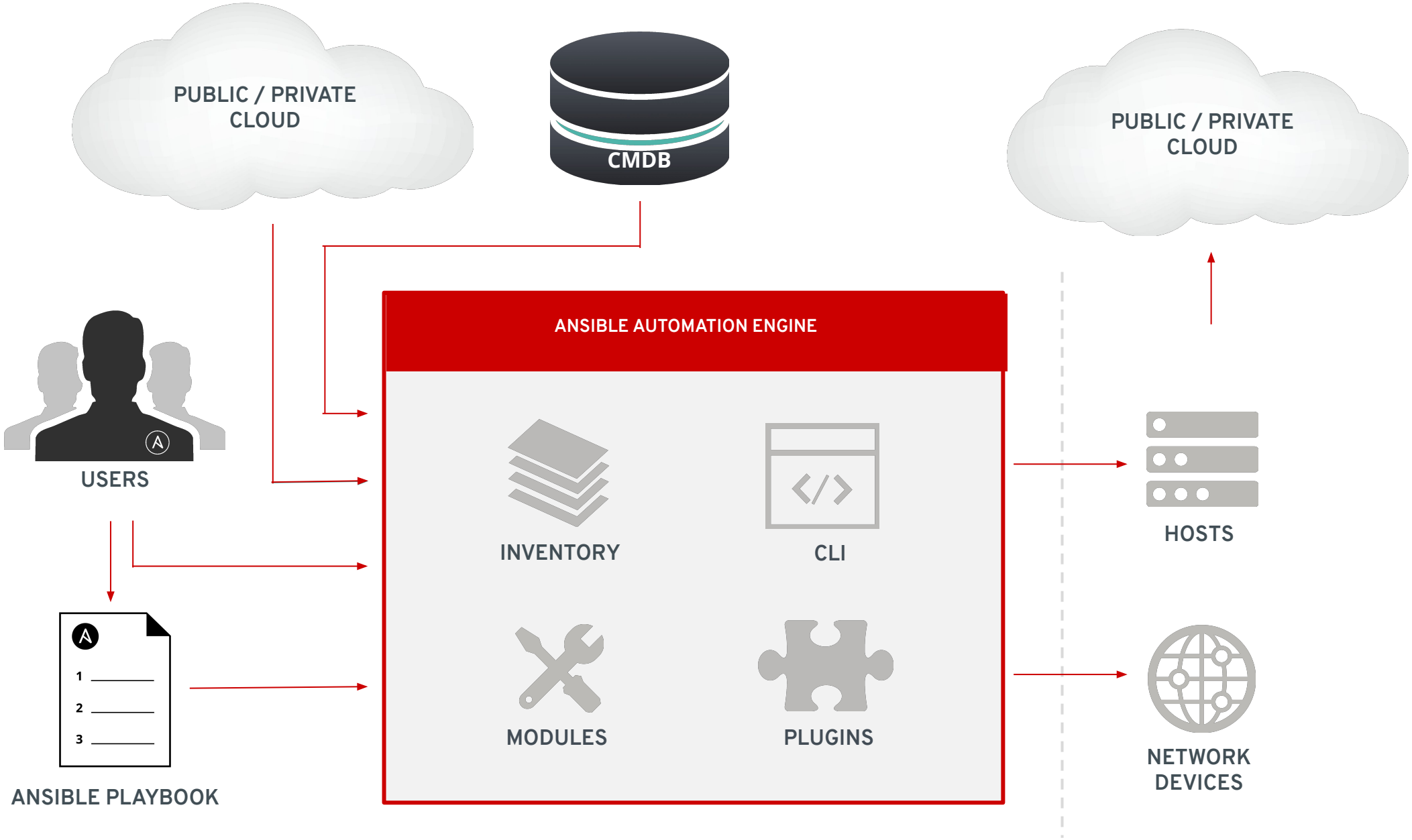
Security

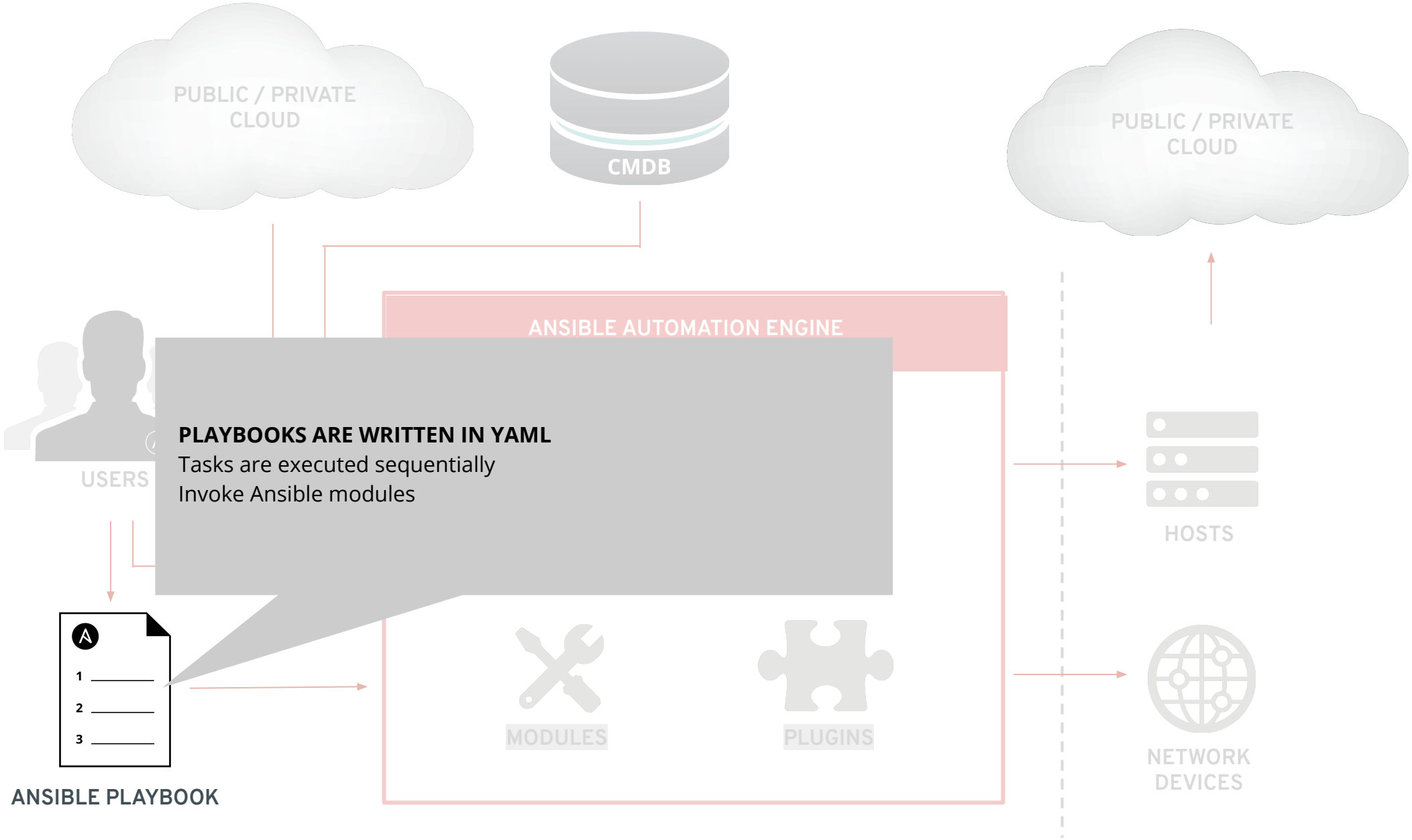
QRadar
Splunk
Snort
Check Point
Fortinet
Cisco FTD
Cyberark
+more

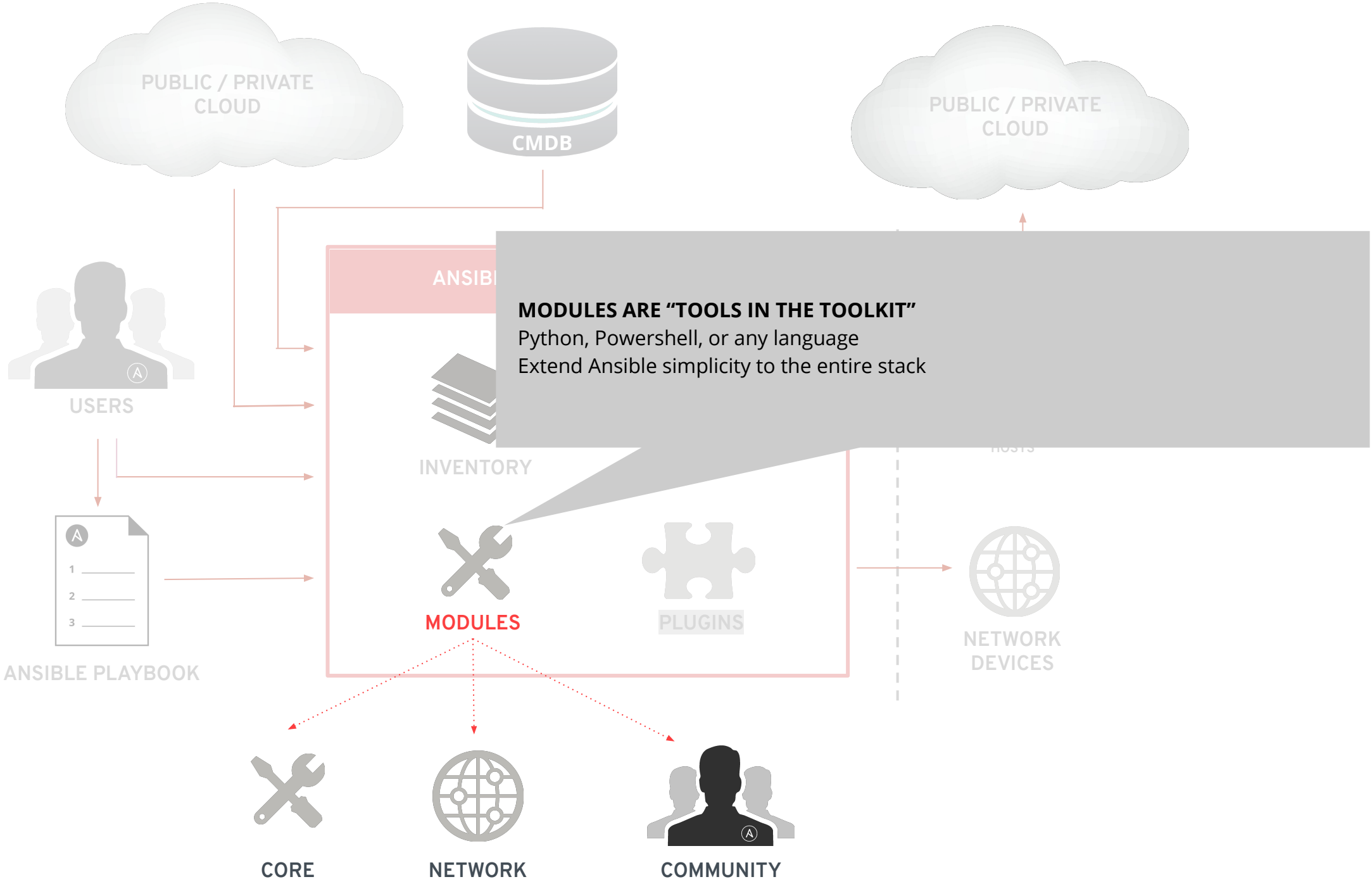
Monitoring

Dynatrace
Airbrake
BigPanda
Datadog
LogicMonitor
Nagios
New Relic
PagerDuty
Sensu
StackDriver
Zabbix
+more

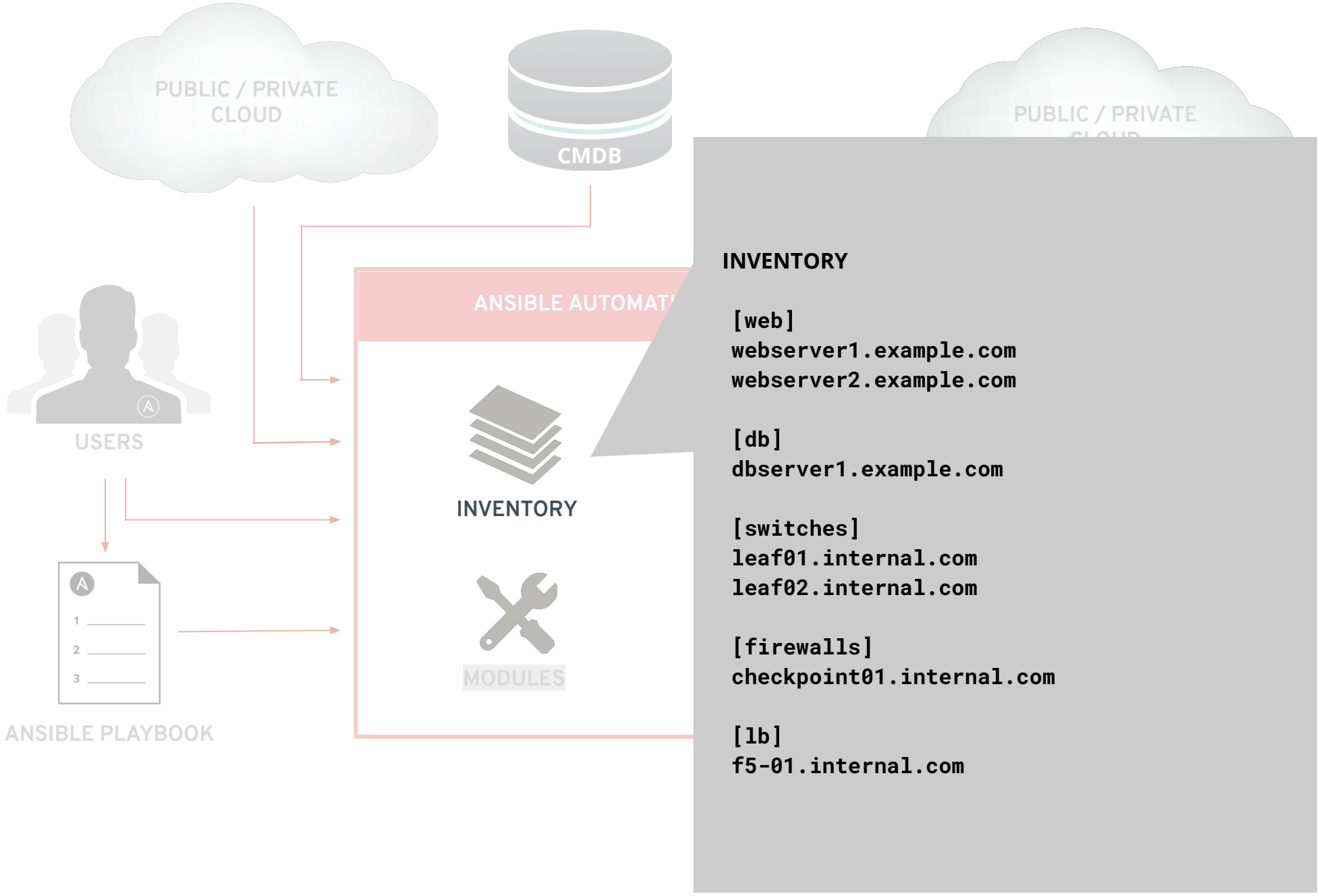
Red Hat

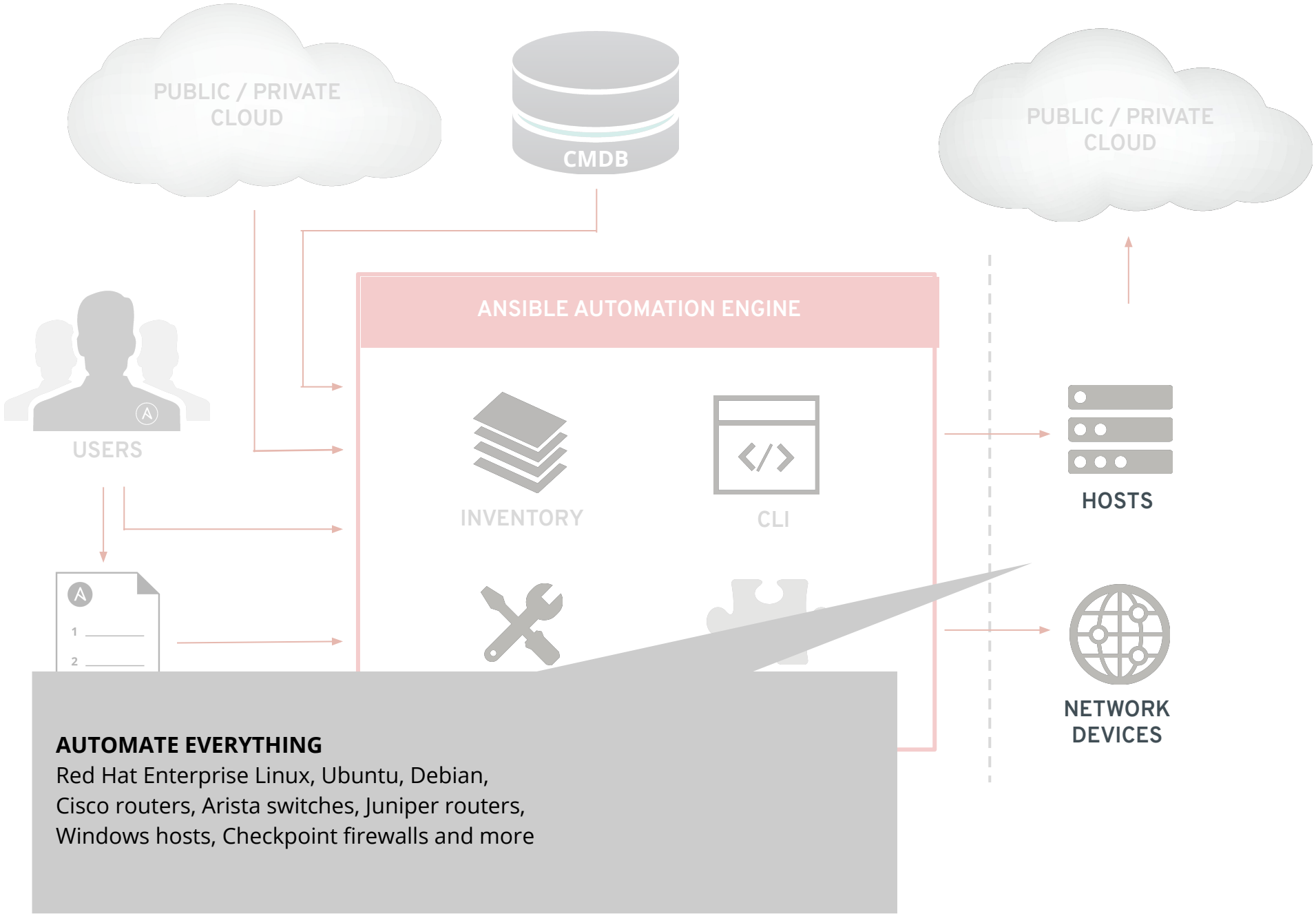

PUBLIC / PRIVATE CLOUD
CMDB
PUBLIC / PRIVATE CLOUD
ANSIBLE AUTOMATION ENGINE
USERS
INVENTORY
CLI
HOSTS
1
2
3
MODULES
PLUGINS
NETWORK DEVICES
ANSIBLE PLAYBOOK


PUBLIC / PRIVATE CLOUD
CMDB
PUBLIC / PRIVATE CLOUD
ANSIBLE AUTOMATION ENGINE
USERS
PLAYBOOKS ARE WRITTEN IN YAML
Tasks are executed sequentially
Invoke Ansible modules
HOSTS
1
2
3
MODULES
PLUGINS
NETWORK DEVICES
ANSIBLE PLAYBOOK



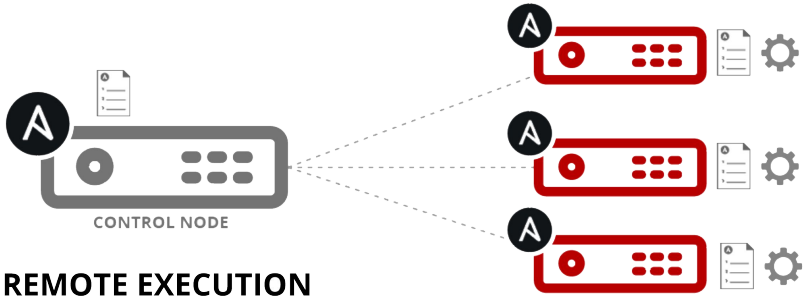
MODULES ARE "TOOLS IN THE TOOLKIT"
Python, Powershell, or any language
Extend Ansible simplicity to the entire stack


PUBLIC / PRIVATE CLOUD
CMDB
PUBLIC / PRIVATE
ANSIBLE AUTOMATI
USERS
INVENTORY
MODULES
ANSIBLE PLAYBOOK
INVENTORY
[web]
webserver1.example.com
webserver2.example.com
[db]
dbserver1.example.com
[switches]
leaf01.internal.com
leaf02.internal.com
[firewalls]
checkpoint01.internal.com
[lb]
f5-01.internal.com


PUBLIC / PRIVATE CLOUD
CMDB
PUBLIC / PRIVATE CLOUD
ANSIBLE AUTOMATION ENGINE
USERS
INVENTORY
CLI
HOSTS
NETWORK DEVICES
AUTOMATE EVERYTHING
Red Hat Enterprise Linux, Ubuntu, Debian,
Cisco routers, Arista switches, Juniper routers,
Windows hosts, Checkpoint firewalls and more

How Ansible Linux Automation works

Module code is copied to the managed node, executed, then removed



LINUX HOSTS

Section 1
Introduction to Ansible Security Automation Basics

Red Hat

Exercise 1.1

Topics Covered:

- How Ansible works
- The lab infrastructure

Red Hat
Ansible
Automation

Red Hat

Ansible Security - What Is It?

Ansible Security Automation is our expansion deeper into the security use case. The goal is to provide a more efficient, streamlined way for security teams to automate their various processes for the identification, search, and response to security events. This is more complex and higher-value than the application of a security baseline (PCI, STIG, CIS) to a server.

Ansible Security Automation is a supported set of Ansible modules, roles and playbooks designed to unify the security response to cyberattacks.

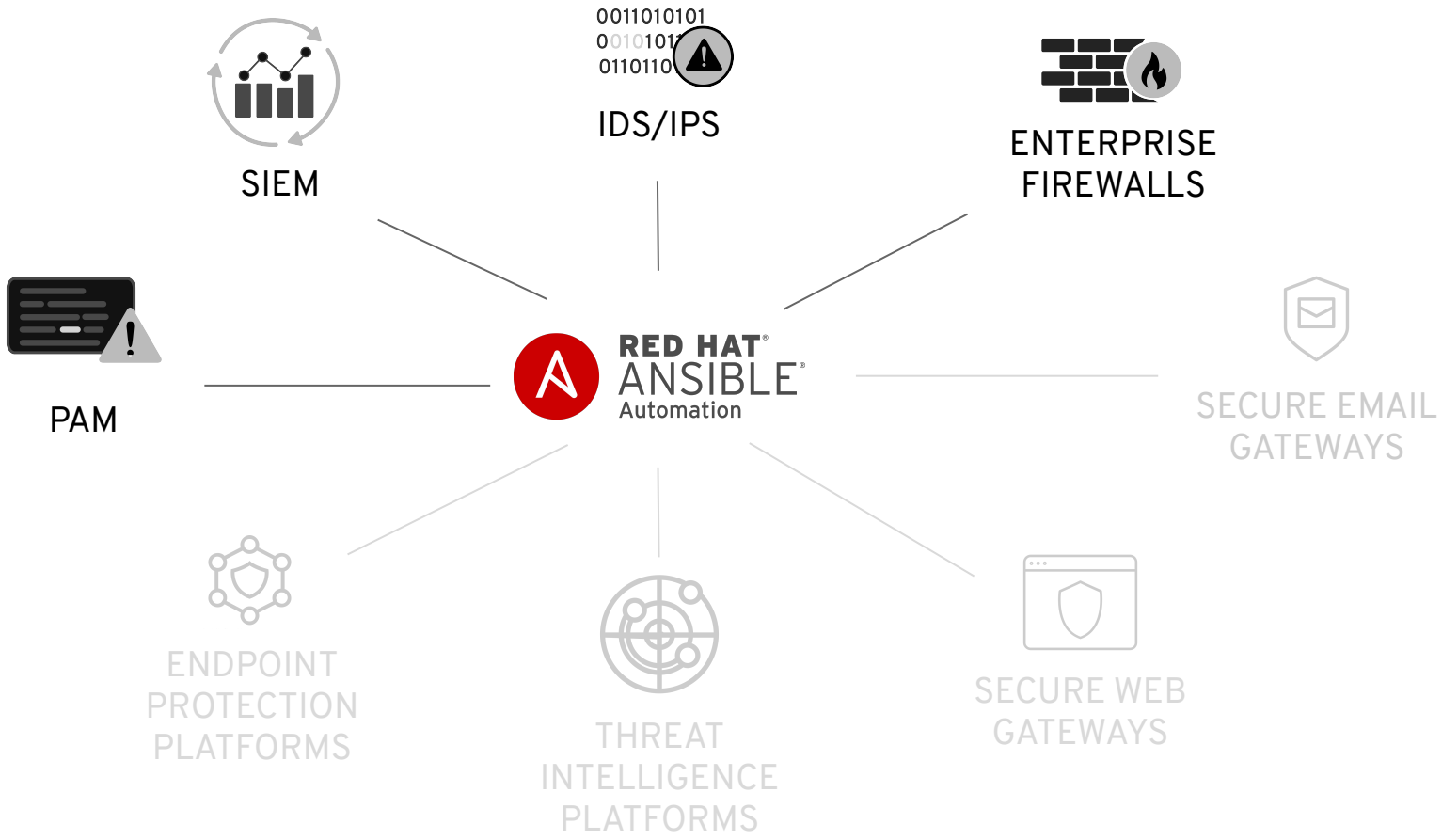
Is It A Security Solution?

No. Ansible can help Security teams “stitch together” the numerous security solutions and tools already in their IT environment for a more effective cyber defense.

By automating security capabilities, organizations can better unify responses to cyberattacks through the coordination of multiple, disparate security solutions, helping these technologies to act as one in the face of an IT security event.

Red Hat will not become a security vendor, we want to be a security enabler.

Ansible Security Automation



Red Hat

In this exercise: Verify Access

- Follow the steps to access environment
- Use the IP provided to you, the script only has example IPs
- Which editor do you use on command line?
  If you don’t know, we have a short intro

Ansible Inventory

- Ansible works against multiple systems in an inventory
- Inventory is usually file based
- Can have multiple groups
- Can have variables for each group or even host

Your inventory

- Contains all machines of your environment
- Setup up just for you, individually
- Note your individual IP addresses for each machine - often in the script you need to replace example IP addresses with your individual ones

Your inventory

```
[all:vars]
ansible_user=student1
ansible_ssh_pass=ansible
ansible_port=22

[control]
ansible ansible_host=22.33.44.55 ansible_user=ec2-user private_ip=192.168.2.3

[siem]
qradar ansible_host=22.44.55.77 ansible_user=admin private_ip=172.16.3.44
ansible_httpapi_pass="Ansible1!" ansible_connection=httpapi ansible_httpapi_use_ssl=yes
ansible_httpapi_validate_certs=False ansible_network_os=ibm.qradar.qradar

[ids]
snort ansible_host=33.44.55.66 ansible_user=ec2-user private_ip=192.168.3.4

[firewall]
[...]
```

Configuration File

- Basic configuration for Ansible
- Can be in multiple locations, with different precedence
- Here: `.ansible.cfg` in the home directory
- Configures where Ansible can find the inventory - and where you find the inventory

Red Hat
Ansible
Automation

Exercise Time - Do Exercise 1.1 Now In Your Lab Environment!

Red Hat

Exercise 1.2

Topics Covered:

- Check Point Next Generation Firewall
- Access via Windows + SmartConsole
- Example interaction via Ansible
- Verify results in the UI

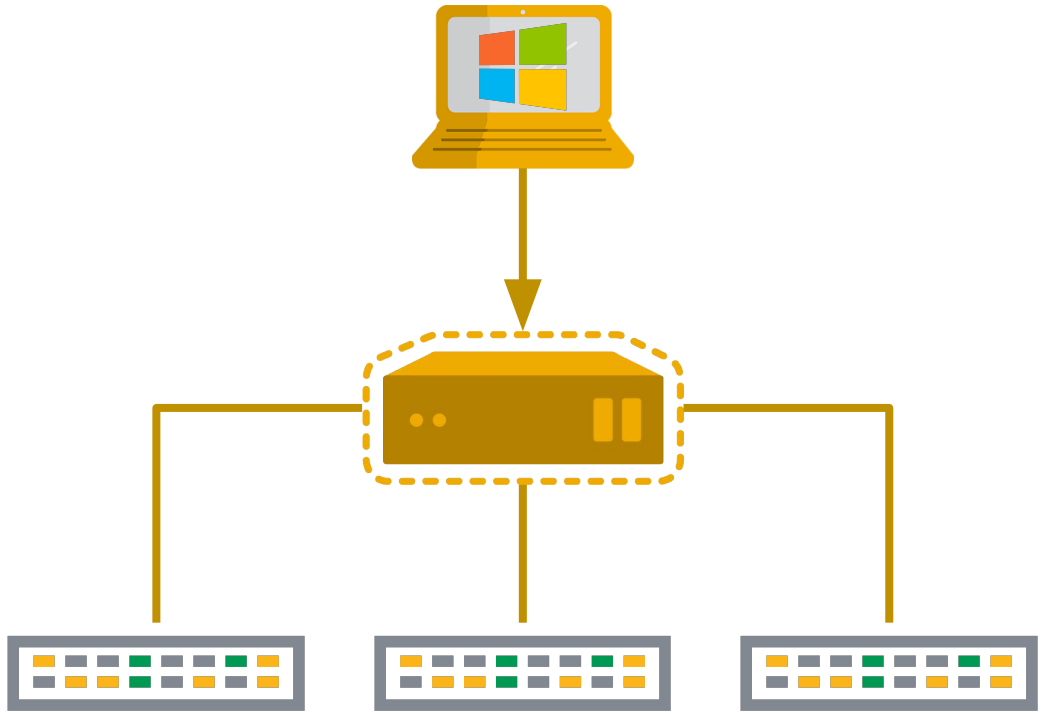
Red Hat Ansible Automation

Red Hat

Accessing And Managing Check Point Next Generation Firewalls

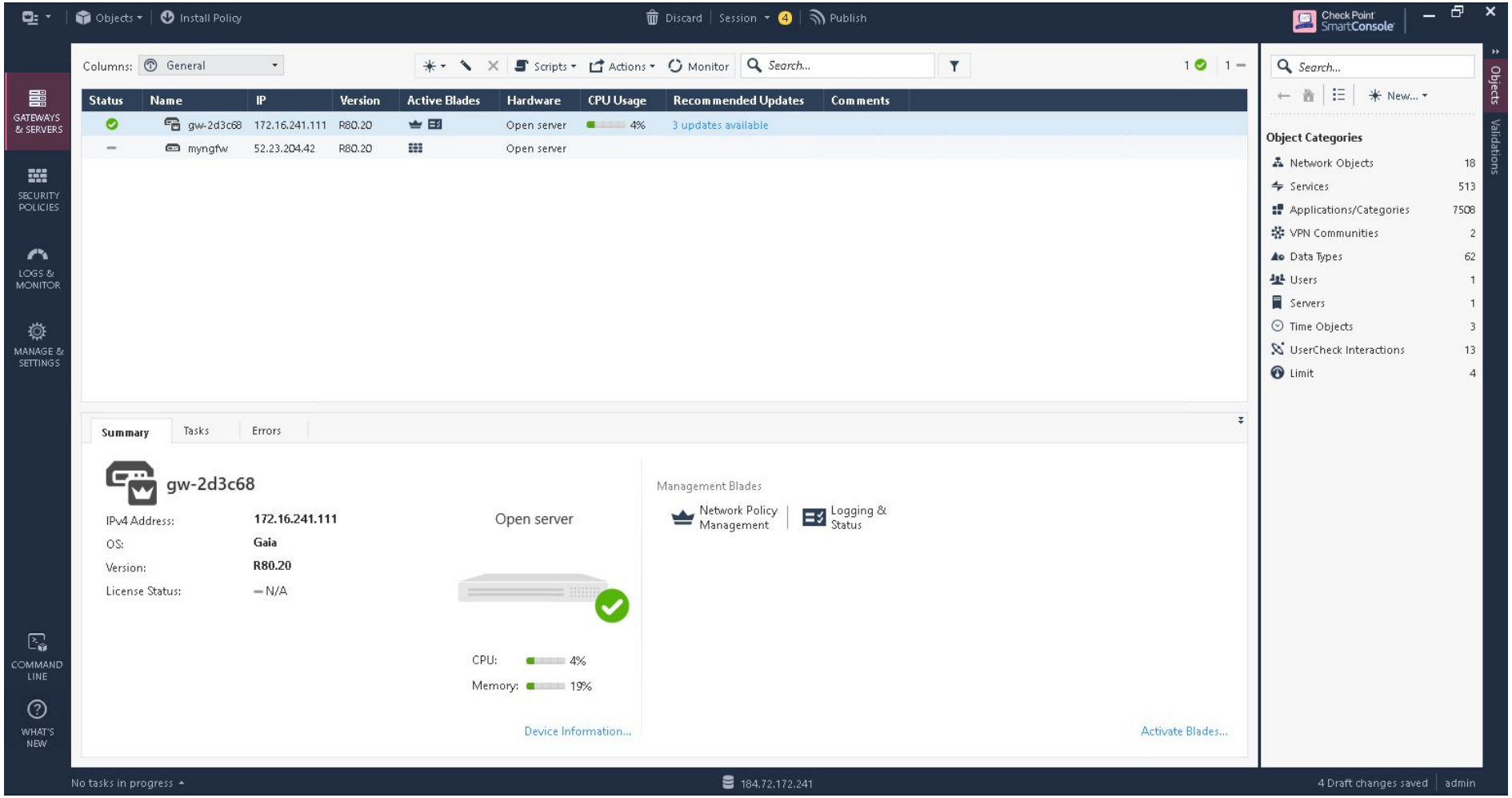
- Access only to central management server
- Via Windows management software, “SmartConsole”
- Automation: HTTP REST API

Lab students: via generic RDP client or RDP-HTML5 client



Red Hat

First Check Point Management Server Login



Red Hat

Run the first playbook

- Playbook is basically list of tasks
- Each task is using a module
- Roles: way to group tasks in re-usable way

An Ansible Play in an Ansible Playbook

A play

Another play

```
- hosts: db
  vars:
    software:
      - mariadb-server
  roles:
    - install_wordpress_db

- hosts: web
  vars:
    software:
      - httpd
      - php
  roles:
    - install_wordpress_web
```

An Ansible Play (Common Elements)

This is not an exhaustive list, but contains most of the elements you will commonly see in an Ansible play.

Connections:

`hosts`	The declarative list of hosts or groups against which this play will run.
`connection`	Allows you to change the connection plugin used for tasks to execute on the target.
`port`	Used to override the default port used in a connection.
`remote_user`	User to define / override which user is connecting to the remote system
`become`	Boolean that controls if privilege escalation is used or not on Task execution. (also **`become_flags, become_user, become_method`**)

An Ansible Play (Common Elements)

This is not an exhaustive list, but contains most of the elements you will commonly see in an Ansible play.

Information Handling:

`name`	Identifier. Can be used for documentation, in or tasks/handlers.
`gather_facts`	Boolean (default **`yes`**) allows the bypass of fact gathering. This can speed up connection time where facts are not needed in a playbook. This refers to the content retrieved by the setup module.
`no_log`	Boolean that controls information disclosure and logging.
`ignore_errors`	Boolean. When set to **`yes`**, errors will be ignored unless absolutely fatal to the playbook execution
`check_mode`	Also known as “dry run” mode, will evaluate but not execute. For modules that support check mode, the module will report the expected result without making any changes as a result of the tasks.

An Ansible Play (Common Elements)

This is not an exhaustive list, but contains most of the elements you will commonly see in an Ansible play.

Inventory Handling:

`order`	Controls the sorting of hosts as they are used for executing the play. Possible values are inventory (default), sorted, reverse_sorted, reverse_inventory and shuffle.

Variable Handling:

`vars`	Dictionary/map of variables
`vars_files`	List of files that contain vars to include in the play.
`vars_prompt`	list of variables to prompt for on launch.

An Ansible Play (Common Elements)

This is not an exhaustive list, but contains most of the elements you will commonly see in an Ansible play.

Task Handling:

pre_tasks	A list of tasks to execute before roles.
roles	List of roles to be imported into the play
tasks	Main list of tasks to execute in the play, they run **after roles** and **before post_tasks**.
post_tasks	A list of tasks to execute after the tasks section.
handlers	Also known as “dry run” mode, will evaluate but not execute. For modules that support check mode, the module will report the expected result without making any changes as a result of the tasks.

Common Ansible Play Elements: Hosts

```
- name: install a LAMP stack
  hosts: web,db,appserver01
  become: yes
  vars:
    my_greeting: Welcome to my awesome page
    favorite_food: fried pickles

  roles:
    - install_lamp_elements

  tasks:
  - name: write the index file
    copy:
      content: {{ my_greeting }}.  Enjoy some {{ favorite_food }}"
      dest: /var/www/html/index.html
    notify: reload_apache

  handlers:
  - name: reload_apache
    service:
      name: httpd
      state: reloaded
```

Ansible Tasks Using Modules:

```
---
tasks:
- name: Ensure httpd package is present
  yum:
    name: httpd
    state: latest

- name: Ensure latest index.html file is present
  copy:
    src: files/index.html
    dest: /var/www/html/

- name: Restart httpd
  service:
    name: httpd
    state: restart
```

Running an Ansible Playbook:

The many colors of Ansible

```
A task executed as expected, no change was made.

A task executed as expected, making a change

General text information and headers

A conditional task was skipped

A bug or deprecation warning

A task failed to execute successfully
```

Running an Ansible Playbook:

```
[user@ansible] $ ansible-playbook apache.yml

PLAY [webservers] ***********************************************************

TASK [Gathering Facts] ******************************************************
ok: [web2]
ok: [web1]
ok: [web3]

TASK [Ensure httpd package is present] **************************************
ok: [web2]
ok: [web1]
ok: [web3]

TASK [Ensure latest index.html file is present] *****************************
ok: [web2]
ok: [web1]
ok: [web3]

TASK [Restart httpd] ********************************************************
ok: [web2]
ok: [web1]
ok: [web3]

PLAY RECAP ******************************************************************
webservers                 : ok=3    changed=3 unreachable=0   failed=0
```

The “Setup” module

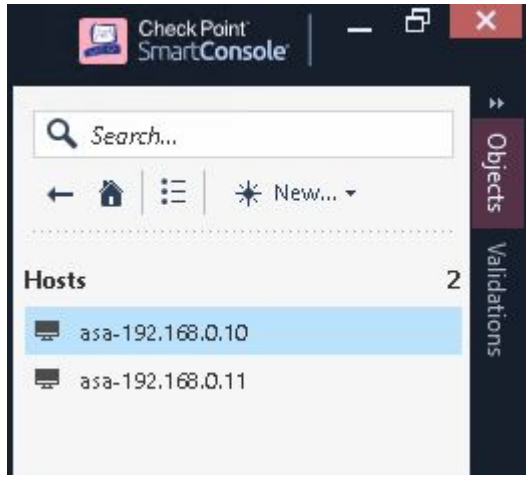
The “yum” module

The “copy” module

The “service” module

Verify Results in UI

- Check network objects for added hosts
- Check policies for added policy



Red Hat

Roles

- A way to load tasks, handlers, and variables from separate files
- Roles group content, allowing easy sharing of code with others
- Roles make larger projects more manageable
- Roles can be developed in parallel by different people

There are pre-built roles for Check Point interaction available.

Role structure

- Defaults: default variables with lowest precedence (e.g. port)
- Handlers: contains all handlers
- Meta: role metadata including dependencies to other roles
- Tasks: plays or tasks
  Tip: It's common to include tasks in main.yml with "when" (e.g. OS == xyz)
- Templates: templates to deploy
- Tests: place for playbook tests
- Vars: variables (e.g. override port)

```
user/
├── defaults
│   └── main.yml
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml
```

Ansible Galaxy

Sharing Content

Community

Roles, and more

How To Install a Role

- Ansible Galaxy command
- Downloads roles from central Galaxy
- Also our roles written as part of the security initiative

```
$ ansible-galaxy install ansible_security.acl_manager
```

Red Hat
Ansible
Automation

Exercise Time - Do Exercise 1.2 Now In Your Lab Environment!

Red Hat

Exercise 1.3

Topics Covered:

- Snort rules
- Running a playbook interacting with Snort

Red Hat Ansible Automation

Red Hat

Snort - Network Intrusion Detection & Prevention System

- Real time traffic analysis and packet logging on IP networks
- Content search and matching
- Service running on possible targets
- in lab: RHEL instance, victim
- Configuration based on rules
- Access and automation: via SSH

Snort Rules

BASIC OUTLINE OF A SNORT RULE

```
[action][protocol][sourceIP][sourceport] -> [destIP][destport] ( [Rule options] )
```

Rule Header

RULE HEADER

The rule header contains the rule's action, protocol, source and destination IP addresses and netmasks, and the source and destination ports information.

alert **Action to take (option)** The first item in a rule is the rule action. The rule action tells Snort what to do when it finds a packet that matches the rule criteria (usually alert).

tcp **Type of traffic (protocol)** The next field in a rule is the protocol. There are four protocols that Snort currently analyzes for suspicious behavior - TCP, UDP, ICMP, and IP.

$EXTERNAL_NET **Source address(es)** variable or literal

$HTTP_PORTS **Source port(s)** variable or literal

-> **Direction operator** The direction operator -> indicates the orientation of the traffic to which the rule applies.

$HOME_NET **Destination address(es)** variable or literal

any **Destination port(s)** variable or literal

EXAMPLE

Rule Header

```
alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any
```

Message

```
msg: "BROWSER-IE Microsoft Internet Explorer
CacheSize exploit attempt";
```

Flow

```
flow: to_client,established;
```

Detection

```
file_data;
    content:"recordset"; offset:14; depth:9;
    content:".CacheSize"; distance:0; within:100;
    pcre:"/CacheSize\s*=\s*/";
    byte_test:10,>,0x3ffffffe,0,relative,string;
```

Metadata

```
policy max-detect-ips drop, service http;
```

References

```
reference:cve,2016-8077;
```

Classification

```
classtype: attempted-user;
```

Signature ID

```
sid:65535;rev:1;
```


Source: https://www.snort.org/documents/snort-rule-infographic

Role To Change Rules

- We have a role to change rules on Snort
- Takes care of service reloading, etc.
- Verification of changes:
  - file system entry
  - another role

Red Hat
Ansible
Automation

Exercise Time - Do Exercise 1.3 Now In Your Lab Environment!

Red Hat

Exercise 1.4

Topics Covered:

- Understanding QRadar
- Collections

Red Hat
Ansible
Automation

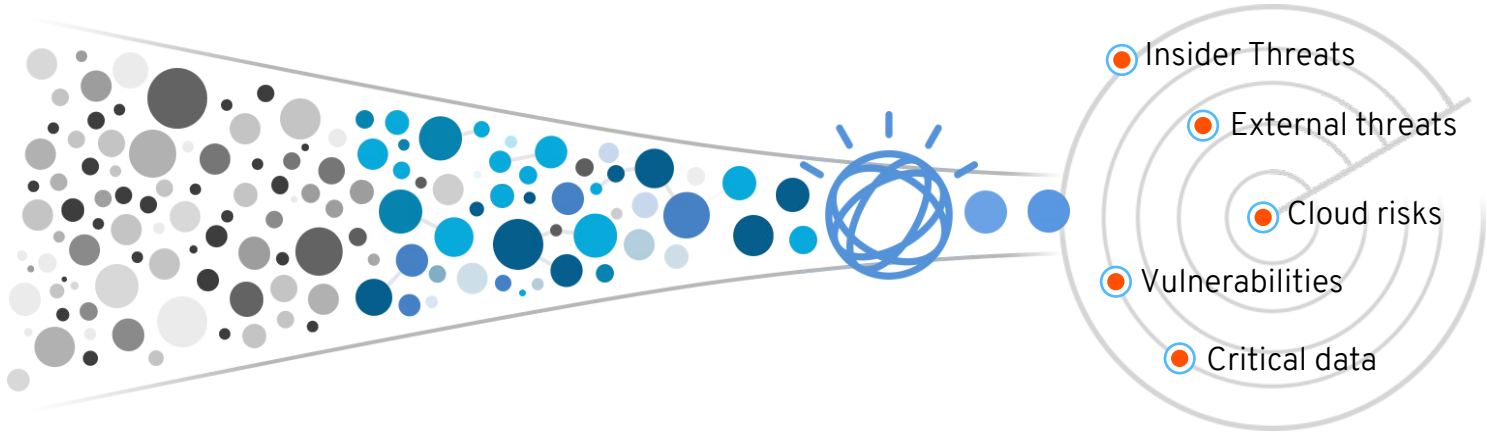
Red Hat

IBM QRadar

Address most important security challenges

Complete Visibility	Prioritized Threats	Automated Investigations	Proactive Hunting

Endpoints
Network activity
Data activity
Users and identities
Threat intelligence
Configuration information
Vulnerabilities and threats
Application activity
Cloud platforms

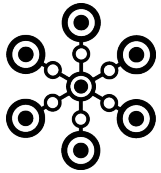


Red Hat

IBM QRadar: Automate Intelligence

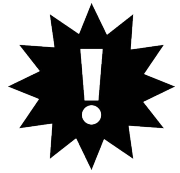
Detect

Known and unknown threats



Connect

Related activity in multi-stage attacks



Prioritize

Business critical events

Investigate

Potential incidents to find root cause faster

Red Hat

QRadar

- SIEM
- Collects & analyses logs
- Can react on specific findings via “Offenses”
- Access via web UI
- Automation via REST API

QRadar

IBM QRadar Security Intelligence - Community Edition

Dashboard | Offenses | Log Activity | Network Activity | Assets | Reports | System Time: 2:15 PM

Show Dashboard: Threat and Security Monitoring | New Dashboard | Rename Dashboard | Delete Dashboard | Add Item... | Next Refresh: 00:00:15

Default-IDS / IPS-All: Top Alarm Signatures (Event Count)

Time Series data unavailable at this time.

View in Log Activity

Top Systems Attacked (IDS/IDP/IPS) (Event Count)

Time Series data unavailable at this time.

My Offenses

No results were returned for this item.

Most Severe Offenses

No results were returned for this item.

Most Recent Offenses

No results were returned for this item.

Top Services Denied through Firewalls (Event Count)

Time Series data unavailable at this time.

Flow Bias (Total Bytes)

Time Series data unavailable at this time.

View in Network Activity

Top Category Types

Category	Offenses
Application Query	0
Host Query	0
Network Sweep	0
Mail Reconnaissance	0
Unknown Form of Recon	0

Top Sources

No results were returned for this item.

Red Hat

Collections

- Ansible content to interact with QRadar: provided as collections
- Like roles, but even more powerful
- Can also contain modules, connection plugins and so on

```
$ ansible-galaxy collection install ibm.qradar \
-p ~/.ansible/collections
```

Verification In The UI

IBM QRadar Security Intelligence - Community Edition

Dashboard | Offenses | Log Activity | Network Activity | Assets | Reports | System Time: 4:30 PM

Offenses

- My Offenses
- All Offenses
- By Category
- By Source IP
- By Destination IP
- By Network
- **Rules**

Display: Rules | Group: Select a group... | Groups | Actions | Revert Rule | DDoS | View the IBM App Exchange for more...

Rule Name	Group	Rule Category	Rule Type	Enabled	Response	Event/Flow Count	Offense Count	Origi
DDoS Attack Detected	D\DoS	Custom Rule	Event	True	Dispatch New Event	0	0	Modified
DDoS Events with High Magnitude Become Offen...	D\DoS	Custom Rule	Event	True		0	0	System
Load Basic Building Blocks	System	Custom Rule	Event	True		0	0	System
Potential DDoS Against Single Host (TCP)	D\DoS	Custom Rule	Flow	False	Dispatch New Event	0	0	Modified

Red Hat

Red Hat
Ansible
Automation

Exercise Time - Do Exercise 1.4 Now In Your Lab Environment!

Red Hat

Section 2
Ansible Security Automation Use Cases

Red Hat

Exercise 2.1

Topics Covered:

- Detection and triage of suspicious activities

Red Hat Ansible Automation

Red Hat

Persona & Situation

- Persona:
  - Security analyst
  - your main tool: SIEM
- Situation:
  - informed of app anomaly
  - nede to figure out if good or bad

Red Hat
Ansible
Automation

Exercise Time - Do Exercise 2.1 Now In Your Lab Environment!

Red Hat

Exercise 2.2

Topics Covered:

- Threat hunting

Red Hat
Ansible
Automation

Red Hat

Persona & Situation

- Persona:
  - Security operator
  - your main tool: Firewall
- Situation:
  - you see suspicious traffic hitting the FW
  - decide to whitelist or not

Red Hat
Ansible
Automation

Exercise Time - Do Exercise 2.2 Now In Your Lab Environment!

Red Hat

Exercise 2.3

Topics Covered:

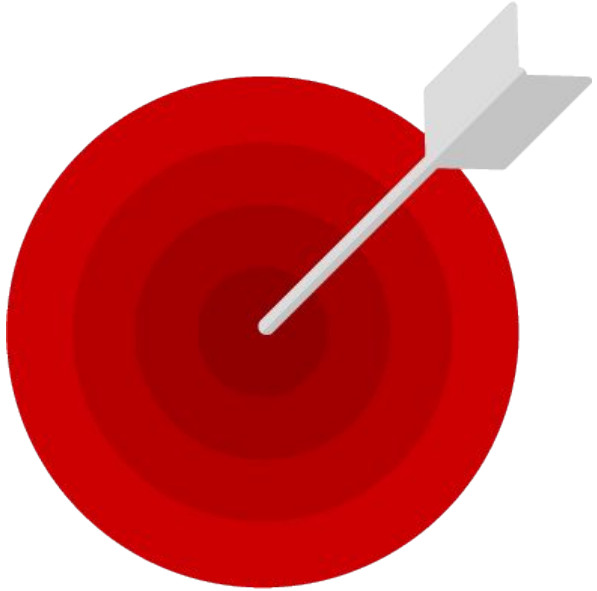
- Incident response

Red Hat
Ansible
Automation

Red Hat

Persona & Situation

- Persona:
  - Security operator
  - your main tool: IDS
- Situation:
  - you see IDS warnings
  - create marker, blacklist



Red Hat
Ansible
Automation

Exercise Time - Do Exercise 2.3 Now In Your Lab Environment!

Red Hat

Exercise 2.4

Topics Covered:

- Wrap uit all up

Red Hat
Ansible
Automation

Red Hat

You Are Done!

You finished the workshop! Just read the final words, and you can soon apply your new knowledge on your own environments!

Red Hat
Ansible
Automation

Exercise Time - Do Exercise 2.4 Now In Your Lab Environment!

Red Hat

Thank you

linkedin.com/company/red-hat

youtube.com/user/RedHatVideos

facebook.com/ansibleautomation

twitter.com/ansible

github.com/ansible

Red Hat