Automating Cloud Infrastructure for network traffic analysis

Brad Palm || Brian Greunke
Outline

• High Level Process
• Terms and Definitions
• Data Movement and Storage
• Building Reusable Infrastructure
• Automating Processes
• Use Cases/Demo
High Level Example

• Get data into cloud

• Pre-process using robust infrastructure and automated processes

• Analyze using robust infrastructure and manual processes
Key Terms

- Provision
- Configuration
- Orchestration
Terms: Provision

- Create new resources
  - Virtual machines
  - Networks
  - Storage
Terms: Configuration Management

- Automate modification of hosts
  - Add software
  - Change settings

1. Install A
2. Install B
...
N. Change setting
Terms: Orchestration

- Automate provisioning and CM

Create xN

Isolated Network
Use Cases

- Network Traffic Analysis
  - Repeatable, deterministic infrastructure
  - Scalable, on-demand infrastructure
  - Remotely accessible, collaborative infrastructure
- Toyota Lean model
Data transfer

- Considerations:
  - Cost / Direction of data flow
  - Time
Data: Transfer

- Into Cloud: (usually) Free
- Out of Cloud: $$$$
## Data Transfer Cost

<table>
<thead>
<tr>
<th>Direction</th>
<th>Cost</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>$0.00</td>
<td>* Snowball may incur fee</td>
</tr>
<tr>
<td>Between</td>
<td>$4.00</td>
<td>Each time</td>
</tr>
<tr>
<td>Object &gt; Block</td>
<td>$0.00</td>
<td>* Intra-region</td>
</tr>
<tr>
<td>Out</td>
<td>$18.00</td>
<td>Each time</td>
</tr>
</tbody>
</table>

* Assuming a 200GB file size for AWS
Data Transfer Time

• Considerations
  – Tool used
  – Location
  – Link quality

** not recommended!!
Data Transfer Time

Time to transfer: **1 TB**

- **T3:** 2.7 days
- **100Mbps:** 1.2 days
- **1000Mbps:** 2.9 hours
Data Transfer Time

```
michael@Winblows-Surface:~$ time aws s3 cp merged1.pcap s3://sharkfest2019/
Completed 422.5 MiB/3.8 GiB (3.6 MiB/s) with 1 file(s) remaining
upload: ./merged1.pcap to s3://sharkfest2019/merged1.pcap
real    18m10.729s
user    1m1.406s
sys     1m14.469s
```
Data Storage

- Object Storage
  - Cheap
  - Collaboration: Easy

- Block Storage
  - More $
  - May be coupled to instance
Data Storage Cost

- **Object (200Gb / month) (no transfer out)**
  - AWS: $4.50
  - Azure: $3.70

- **Block**
  - AWS: $8.60
  - Azure: $10.00
## Compute Costs

<table>
<thead>
<tr>
<th>Instance</th>
<th>$ / Hour</th>
<th>$ / Day</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 vCPU 1 GiB RAM</td>
<td>$0.00*</td>
<td>$0.00*</td>
</tr>
<tr>
<td>2 vCPU 16 GiB RAM</td>
<td>$0.14</td>
<td>$3.36</td>
</tr>
<tr>
<td>16 vCPU 128 GiB RAM</td>
<td>$1.12</td>
<td>$26.88</td>
</tr>
</tbody>
</table>
Live Drawing of VPC & Cloud Concepts!
Use Moloch for Indexing and Analysis

- Requires Moloch and separate instance(s) of Elastic search

1. Provision instance(s) of Elastic Search
2. Provision Moloch instance
3. Configure Elastic Search
4. Configure Moloch
Provision: Terraform

resource "aws_instance" "elastic-search" {
  ami = "ami-b374d5a5"
  instance_type = "r5.2xlarge"
  count = 2
}

resource "aws_instance" "moloch" {
  ami = "ami-b374d5a5"
  instance_type = "t2.medium"
  count = 1
}
Provision: Terraform

[user@host]$ terraform plan

[user@host]$ terraform apply

[user@host]$ terraform destroy
[user@host]$ ansible-playbook -i hosts moloch.yml
Configure: Ansible

# moloch.yml
---

- hosts: elasticsearch
  roles:
    - { role: elasticsearch }

- hosts: moloch
  roles:
    - { role: moloch }
Configure: Ansible

```yaml
# roles/elasticsearch/tasks.yml

---
- name: Amazon Linux - Install Elasticsearch
  become: yes
  yum:
    name: 'elasticsearch'
    state: present
    update_cache: yes
    allow_downgrade: 'yes'
  when: es_use_repository
  notify: restart elasticsearch
```
- name: Amazon Linux - Install Elasticsearch
  become: yes
  yum:
    name: 'elasticsearch'
    state: present
    update_cache: yes
    allow_downgrade: 'yes'
  when: es_use_repository
  notify: restart elasticsearch
More Examples

• Carve large PCAP using tcpdump/tshark
• Analyze large PCAP using Wireshark on a heavy-duty instance
• Parallel process multiple captures using multiple cloud instances
• Build verifiable analysis tools
Parallel Processing

- 7 PCAPs (each day over a week)
- Same processing required for each prior to analysis
- Create 7 instances, pass PCAP to each, process independently, in parallel
Demo Example

• Use Case:
  – Large PCAP
  – Need to carve the PCAP
  – Needs to be done quickly
Demo Example (cont)

• Steps:
  - Move to S3 using “aws-cli” tool
  - Need to carve the PCAP
  - Needs to be done quickly
Carving a large pcap

- ~ 4 Gb
- > 3.6 Million Packets
- Encrypted HTTP captured on trunk port w/ VLAN tags
- A tale of two machines
#!/bin/bash

# Create directory for individual streams
mkdir -p ./streams

# Pull TCP stream numbers from pcap
tshark -r large.pcap -T fields -e tcp.stream > streams.log

# Sort and filter unique TCP stream numbers
cat streams.log | sort -n | uniq > sorted.log

# Extract streams from pcap in parallel
parallel -a sorted.log 'tshark -r large.pcap -Y "tcp.stream == {}" -w ./streams/{}.pcap'
Attempt #1 Local Demo

- This ran for 8hrs
- Never finished the first part of the parsing script
- 2,367 streams were found of the 6.6M streams that were actually there
- Could not complete the job, given the tool!
What Do?
Provision Demo

*Video of provisioning the analysis machine within AWS
Attempt #2 Cloud Demo

- r5.24XL
- 96 vCPUs
- 768GB RAM
- Task took ~2hrs
Demo 2 – Local FAIL

*Trying to have Wireshark open the file on a laptop
Demo 2 Configure

*Video of configuring the cloud analysis machine with Wireshark
Demo 2 – Cloud WIN

*Video of remotely connecting to cloud resource and then successfully opening the large PCAP in Wireshark
Questions???

brian@openonelabs.com
brad@bruteforce.io
linkedin.com/in/bradpalm/
www.bruteforce.io