Five-Ish Ways to Kick Traditional Security’s Ass with Cloud and DevOps”

Rich Mogull
@rmogull
Cloud can be more secure than traditional datacenters.

- The economics are in your favor.
- Cloud architectures can wipe out some traditional security headaches.

This isn’t theory, it’s being done today.

- But only if you understand how to leverage the cloud.

We will show you how.
Not the SaaS you’re looking for

This session is all IaaS and PaaS
For clients to use a cloud provider, they must trust the provider.

This is especially true for anything with a sensitive data or process.

Thus security has to be a top priority for a provider or you won’t use them.

A major breach for a provider that affects multiple customers is an existential event.
Cloud Provider Critical Security Capabilities

- API/admin activity logging
- Elasticity and autoscaling
- APIs for all security features
- Granular entitlements
- Good SAML support
- Multiple accounts per customer
- Software defined networking
- Region/location control
- Nice to have: infrastructure templating/automation
Problem 1: Flat Networks and Data Centers
Segregation is critical but hard

- Segregating networks in a data center is hard, expensive, and often unwieldy.
- It’s hard to isolate application services on physical machines.
- Even using virtual machines has a lot of management overhead.
- Attackers drop in and move North/South in application stacks, and East/West on networks (or both).
Network segregation by default

Granularity of a host firewall with the manageability of a network firewall
Limiting blast radius
To a host or network…

Account

Virtual Network

Subnet

Security Group

Virtual Network

Subnet

Security Group
To a host or network…

- Virtual Network
- Subnet
- Security Group
- Account

Boom
Or an entire “data center”
Or an entire “data center”
Traditional blast radius
Application segregation

- Easier to deploy smaller services
- Easier to isolate
- Can integrate PaaS for “network air gaps”
Cloud “DMZ”
{
  "AWS::TemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "KeyName": {
      "Description": "The EC2 Key Pair to allow SSH access to the instance",
      "Type": "String"
    }
  },
  "Resources": {
    "LabVPC": {
      "Type": "AWS::EC2::VPC",
      "Properties": {
        "CidrBlock": "10.0.0.0/16",
        "InstanceTenancy": "default",
        "EnableDnsSupport": "true",
        "EnableDnsHostnames": "true",
        "Tags": [
          {
            "Key": "Name",
            "Value": "CloudSec Lab"
          },
          {
            "Key": "Group",
            "Value": "Vpclab"
          }
        ]
      }
    },
    "subnet667a7420": {
      "Type": "AWS::EC2::Subnet",
      "Properties": {
        "CidrBlock": "10.0.0.0/24",
        "AvailabilityZone": "us-west-2c",
        "VpcId": {
          "Ref": "LabVPC"
        },
        "Tags": [
          {
            "Key": "Name",
            "Value": "CloudSec Lab Subnet"
          }
        ]
      }
    }
  }
}
Problem 2: Security Code Testing
Complexity Breeds Error
Automate Creation of Master OS Images

- Git
- Jenkins
- Security Tests
- Master Image

- Ops/Server Engineering Requirements
- InfoSec Requirements
- Packer configuration
Demo – Server Image Bakery/Factory with Automated Security Testing

Update the desired configuration of a new master OS image
Build the master image
Test the master image for security controls
Make image available for use
Problem 2(.5): Patch and Change Management
Managing Patches and Change

- Nothing we deploy is consistent.
- Even when we become consistent, it’s hard to patch live stuff without breaking things.
- Privileged users log into servers and make changes.
- Attackers love persistent servers they can compromise and camp inside.
- Plus, we need to keep the auditors happy.
The Power of Immutable

- Instead of updating, you completely replace infrastructure through automation.
- Can apply to a single server, up to an entire application stack.
- Incredibly resilient and secure. Think “servers without logins”.

Image from: http://tourismplacesworld.blogspot.com/2012/07/uluru.html
How immutable works - auto scaling

Load Balancer

Auto Scale Group
How immutable works - auto scaling
How immutable works - auto scaling

Load Balancer

Auto Scale Group
How immutable works - auto scaling

Vulnerable

Load Balancer

Auto Scale Group

Patched

a

b

c
How immutable works- auto scaling

Vulnerable

Auto Scale Group

Load Balancer

Patched
How immutable works - auto scaling

Vulnerable

Patched

Auto Scale Group

Load Balancer
How immutable works- auto scaling

Load Balancer

Vulnerable

Patched

Auto Scale Group
Demo

Rolling update of 40 instances in 4 minutes with 0 downtime.
Immutable Infrastructure
Automate with DevOps and Continuous Deployment
Immutable Infrastructure
Immutable Infrastructure

Template A:

Template B:
Immutable Infrastructure

Template A:

Template B:
Immutable Infrastructure

Template B:
Immutable Infrastructure

Template B:
Problem 3: Server and Network Hardening
Let your PaaS do the work

- We deploy many MANY core components to deliver applications.
  - Load balancers, databases, message queues, and more.
- It takes a lot of effort to keep these secure and up to date at scale.
- Each piece is yet more attack surface.
PaaS providers can’t afford a preventable security failure.
- Including letting things get out of date.
- Many types of PaaS can’t rely on normal networking.
- Instead you access them via API.
- This creates an opportunity to “air gap” parts of your application.
- Kill off network attack paths (doesn’t help with logic flaws)
Network attack path?
PaaS Air Gap

No direct network connection

Internet → Elastic Load Balancer → Web Servers → Simple Queue Service → Processing Servers → DynamoDB
Problem 4: Agile Security Operations
Attackers are automated, we are mostly manual.

Our tools have been poor.

We lack trustable security automation and thus need to rely on a “Meat Cloud”

In cloud, APIs are mandatory. We can write code to automate and orchestrate, even across products and services.
Code without coding

- Work with your devs to build a library of building blocks
- Learn just enough to glue it together
- Build some core scrips
- Mix and match the blocks
Meet SecuritySquirrel, the first warrior in the Rodent Army (apologies to Netflix).

The following tools are written by an analyst with a Ruby-for-Dummies book.

Automated security workflows spanning products and services.
Incident Response

1. Detect Compromise
2. Pull server information (If you have it)
3. Quarantine
4. Image
5. Analyze
6. Recover

= Hours!

Each step is manual, and uses a different set of disconnected tools.
1. Pull metadata
2. Quarantine
3. Swap control to security team
4. Identify and image all storage
5. Launch and configure analysis server
6. Can re-launch clean server instantly
Stateless Security

- Security normally relies on scanning and checking databases.
- With cloud we are completely integrated into the infrastructure and platforms.
  - The cloud controllers have to see everything to manage everything, there is no Neo running around.
- Instead of scanning, we can directly pull state.
  - And then use it for security
Identify Unmanaged Servers (for the audit)

1. Scan the network
2. Scan again and again for all the parts you missed
3. Identify all the servers as best you can
4. Pull a config mgmt report
5. Manually compare results
Get list of all servers from cloud controller (can filter on tags/ OS/etc).

- Single API call

Get list of all servers from Chef

- Single API call

Compare in code
Event Driven Security

- Cloud providers are creating hooks to trigger actions based on events inside the cloud.
- We can use these for near-instant security reactions.
Problem 5: Response Time
Self-Healing Infrastructure (yes, for real)

- Change a security group
- Event Recorded to CloudTrail
- Passed to CloudWatch Log Stream
- Triggers an CloudWatch Event
- Lambda Function analyzes and reverses

<table>
<thead>
<tr>
<th>Type</th>
<th>Protocol</th>
<th>Port Range</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSH</td>
<td>TCP</td>
<td>22</td>
<td>68.2.174.98/32</td>
</tr>
<tr>
<td>HTTP</td>
<td>TCP</td>
<td>80</td>
<td>0.0.0.0/0</td>
</tr>
</tbody>
</table>
Demo

Watch a security group heal itself in less than 10 seconds
Conclusions

- Cloud and DevOps are more security than traditional infrastructure and operations
  - Thanks to automation and orchestration that create consistency and agility
  - But only if you go native
- Start with controlling blast radius and adopting immutable infrastructure
- Then integrate security code and server testing
- Go serverless to reduce attack surface and make your cloud provider do the work
- Adopt software defined security for security at the speed of cloud and DevOps
Five-Ish Ways to Kick Traditional Security’s Ass with Cloud and DevOps

Rich Mogull
@rmogull