A Virtual and Software-Defined Security Architecture Workshop

Dave Shackleford
Sr. Faculty
SANS Institute
@daveshackleford
Introduction

- The world’s gone virtual! Most IT environments are heavily virtualized and are starting to leverage SDN and API-driven components.
- Major SDN tools and vendor products have emerged.
- Today, as more and more organizations are virtualized and moving to SDN, we have:
  - Different architectural models
  - Different components
  - A totally new “security stack”
Virtual Networking: NFV and SDN

- Network Functions Virtualization (NFV) decouples network functions from dedicated hardware devices
  - Network services (routers, firewalls, load balancers, etc.) can now be hosted on virtual machines

- SDN is an architectural model that offers network virtualization and programmability
  - SDN abstracts the network control plane from the data plane
  - Some definitions are less focused on decoupling the planes, and more on APIs and integration
The Big Picture: Software-Defined Security

- Increasingly, we see Security as Code taking shape
- Security process automation needs to move at DevOps speed
- Less ownership, visibility, control
- Distributed “Workloads” – virtual, abstract, transient
- Large, flat, shared networks
- More and faster changes, with less control
Defining the New Security Stack
The Technology Stack: Hypervisors

- Hypervisors are a low layer of the stack
- Private clouds require patching, configuration and access management
- Public clouds: Little to no visibility or control
The Technology Stack

• Confidentiality and integrity of data moving from your environment to CSP, and within CSP
• Access controls to resources
• Network protection (Layers 2-7)
• Availability
• Segregation/zones/domains

<table>
<thead>
<tr>
<th>Network</th>
</tr>
</thead>
</table>

| Hypervisor    |
The Technology Stack

- Look at encryption controls for data at rest and in transit
- Cloud access gateways
- CASBs for data monitoring and content control
- DLP within SaaS environments
The Technology Stack

- OS hardening + access control
- Anti-malware and whitelisting
- Logging and monitoring

<table>
<thead>
<tr>
<th>Operating Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
</tr>
<tr>
<td>Network</td>
</tr>
<tr>
<td>Hypervisor</td>
</tr>
</tbody>
</table>
# The Technology Stack

- Appsec + WAFs + CASBs + IAM + ???

<table>
<thead>
<tr>
<th>Application Logic + Presentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Systems</td>
</tr>
<tr>
<td>Data</td>
</tr>
<tr>
<td>Network</td>
</tr>
<tr>
<td>Hypervisor</td>
</tr>
</tbody>
</table>
A Basic Reference Architecture

Hypervisor security: No control
Network security: Layers 3-7, NFV+SDN
Data Security: Encryption, lifecycle, DLP
OS Security: Config+Patching
Apps: Code control + RBAC

Hypervisor security: Full control
Network security: Layers 2-7, NFV+SDN
Data Security: Encryption, lifecycle, DLP
OS Security: Config+Patching
Apps: Code control + RBAC
Hypervisor Security
Virtualization platforms are often in use within cloud environments.

Once you have an idea of your technology and some of the gaps, take a look at:

- Operational practices and gaps
- Gaps in policy and standards (example: hardening guidance for hypervisors)
- Gaps in security products within the virtual environment (example: antivirus tools or identity management)

Your first area of focus should be the main components: hypervisors, virtual networks, storage and management tools.
Hypervisor Security Controls

- Foundational controls
  - NTP, SNMP, etc
- Local firewall/network access controls
- Hardening and configuration
- Users and Groups
- Patching
- Logging and Monitoring
- SELinux and/or multitenant isolation measures
Virtual Network + SDN Security
Network Security

- **Critical areas to focus on for security:**
  - Ensuring confidentiality and integrity of data movement
  - Access controls to resources
  - Network protection (Layers 2-7)
  - Availability
  - Segregation/zones/domains
With NFV and SDN, the concept of “service chaining” in security is important

A single platform can accommodate:
- Anti-malware
- Network access controls
- Anomaly detection
- Intrusion prevention
- Etc
NFV Security Considerations

- NFV platforms have many capabilities, but new risks:
  - Resource sharing, role/privilege models, encryption exposure, etc.

- Look into the following:
  - Vendor code review and security
  - APIs exposed and used by NFV platforms
  - Security configuration settings and patching for NFV solutions

- Perform regular scans and assessments of NFV tools and components
Controllers are the “brains” of SDN
- Centralized
- Programmable
- Attackable

Focus on: patching and basic service security (HTTPS, SSH)

Focus on: role-based access and authentication/authorization
It’s time to shift...

- From THIS:

- To THIS:

Type: "AWS::EC2::SecurityGroupIngress"
Properties:
  - CidrIp: String
  - CidrIpv6: String
  - FromPort: Integer
  - GroupId: String
  - GroupName: String
  - IpProtocol: String
  - SourceSecurityGroupName: String
  - SourceSecurityGroupId: String
  - SourceSecurityGroupOwnerId: String
  - ToPort: Integer
New architecture options: Focus on microsegmentation

- Each cloud instance adopts a “zero trust” policy model for granular network interaction controlled at the virtual machine NIC(s)
- Network policy can “travel” with each instance
Management + Storage Security
Management tools and bastion hosts are critical aspects of a secure software-defined architecture.

Controls include:
- Platform hardening
- Access restriction and role-based access control
- Continuous logging+monitoring

Common tools include VMware vCenter, Microsoft SCVMM, and Citrix XenCenter.

Cloud management tools like OpenStack are common, too.
For software-defined storage instantiation, there are three major considerations:

- Type of storage and tactical aspects (size, scale, location, etc)
- Encryption or other data protection controls
- Access controls

Logging and monitoring of storage access and operations is critical, as well
Example: S3 Bucket Creation

1. Create an S3 bucket at the command line:
   
   ```bash
   $ aws s3 mb s3://bucket-name
   ```

2. Add bucket access controls:
   
   ```bash
   $ aws s3api put-bucket-acl --bucket BucketName --grant-full-control 'emailaddress="dave@sans.org"' --grant-read 'uri="http://acs.amazonaws.com/groups/global/AllUsers"'
   ```

3. Encryption in REST headers:
   
   ```bash
   x-amz-server-side-encryption
   ```
S3 Logging Policy (example)

```json
{
    "LoggingEnabled": {
        "TargetBucket": "BucketName",
        "TargetPrefix": "BucketLogs/",
        "TargetGrants": [
            {
                "Grantee": {
                    "Type": "AmazonCustomerByEmail",
                    "EmailAddress": "dave@sans.org"
                },
                "Permission": "FULL_CONTROL"
            }
        ]
    }
}
```
Orchestration + Automation
Orchestration: Single Point of Failure?

Reference: http://inthepassing.files.wordpress.com/2010/01/cloud-ref-arch.jpg
Orchestration and Automation Risks

- **Control of and interaction with automation platforms can be very risky**
  - Poor development, scripting, resource design and instantiation
  - System availability issues or resource hijack/compromise
  - Malicious insiders or lack of “least privilege”
  - Vendor lock-in (architecture, language, etc.)
  - Poor authentication/credential management
  - Weak or non-existent integration with security products

- **Configuration management and access control are critical**
Orchestration/Automation Security

- **Orchestration Platforms**
  - Often multi-tiered
  - Focus on code/data repos, master servers, and client configs

- **Databases**
  - Usernames and passwords, config files containing sensitive data

- **Automation platforms**
  - Separate repos used for configuration and resource management
Orchestration/Automation Security

- **Operations teams**
  - Social engineering attacks targeting orchestration and automation teams - more focus on security awareness

- **API calls and logging**
  - Local access and calls of APIs
  - Remote API logging at nodes and infrastructure

- **“Failsafes” – affected platforms and systems**
  - “Deny All” stance and “triggers”/”tipping point” fallbacks
An AWS Example: Pulling it all Together

- Software-defined infrastructure template
- Configuration definition
- Code analysis?
- Logs and event data
- Logs and event data
- Logs and event data
- Logs and event data
- Logs and event data
- Logs and event data
- Logs and event data
- Logs and event data
- Logs and event data
- Logs and event data
Additional Security Considerations
IaaS and PaaS: Focus on config and patch management

- Define configuration items and baselines
- Approve configuration templates and controls
- Embed configuration standards in builds
- Automate patch management as much as possible
IaaS and PaaS: Focus on vulnerability scanning

- Check for scanning products that have been adapted to cloud
  - Some have strong API support and integration
- Also consider host-based assessment
All cloud models: Focus on privilege management

- Carefully limit and control the accounts and privileges assigned to resources
- All users, groups, roles, and privileges should be carefully discussed and designated to resources on a “need to know” basis
- Assign “least privilege” and monitor carefully
- Consider IDaaS options that can mandate strict central control and monitoring (and API integration)
- Check for keys and credentials in code!
Next week you should:
- Evaluate all stack layers and components, and ensure you know what you have

In the first three months following this presentation you should:
- Look at/for software versions of your current hardware security platforms
- Discuss internal use cases for software-defined security and SDN

Within six months you should:
- Possibly have software definitions for system builds and architecture models
- Consider how automation and orchestration of security functions might work in your environment...and to the cloud.