Karthik Muthukrishnan

Automating Cloud Security Governance

Helsinki Metropolia University of Applied Sciences

Master's Degree

Information Technology

Master's Thesis

14 May 2017

Orca Security Ltd.
Exhibit 2190
Wiz v. Orca
IPR2024-01190
Ex. 2190-001

Author(s) Title Number of Pages Date	Karthik Muthukrishnan Automating Cloud Security Governance 57 pages + 3 appendices 14 May 2017
Degree	Master of Engineering
Degree Programme	Information Technology
Instructor(s)	Janne Salonen, Principal Lecturer

Adopting cloud infrastructure in a large scale is a challenging proposition for companies. One of the key challenges is adopting the organization's existing infrastructure and security governance to the cloud operations. Scalability, agility and distributed computing are inherent properties of cloud infrastructure. These precisely are the challenges faced on governing cloud security.

Existing procedures that depended on manual intervention are not feasible when the infrastructure is almost infinitely (compared to human resources at disposal) scalable. Parts of the infrastructure can change abruptly within minutes. How does one deploy audit processes with such agile infrastructure? What if the infrastructure changes even before the audit is complete? How to provide security assurance to higher management while following rapid-release cycles in DevOps mode?

This project is a series of governance learnings, tools prototyping and experimentation done on job. Infrastructure governance policies, procedures and tools were created specifically for the cloud. Parts of the cloud infrastructure such as the OS were customized to meet the governance policies. Asset and identity management were achieved by centralizing cloud service accounts. This enabled a central team to use cloud APIs to manage assets and users. Automated tools were deployed centrally to audit cloud assets and user accounts for security issues.

The results strongly indicate that security automation and self-certification are key components of security governance of cloud and DevOps.

Keywords	security, automation, devops, aws, cloud, agile

Helsinki Metropolia University of Applied Sciences

Contents

1 Introduction 1

1.1 Method and Process 2
1.2 Thesis Structure 3

2 Information Security Governance 4

2.1 Security Strategy 4
2.2 Roles and Responsibilities 4
2.3 Enterprise Architecture 5
2.4 Policies and Guidance 5

3 Amazon Web Services Cloud Computing 8

3.1 Cloud Service Models 9
3.2 Amazon Web Services 9
3.3 Sample Web Service Architecture in AWS 11

4 Agile and DevOps 14

4.1 Agile Methodology 14
4.2 DevOps 14
4.3 Security Governance Must Be Agile 16

5 Security Governance and Process Assessment 17

5.1 Security Strategy 17
5.2 Adopting Security Processes to DevOps and Scrum 19
5.3 Organizational Structure and Responsibilities 20
5.4 Production and R&D Accounts 21
5.5 Policies and Guidance for AWS Projects 22
5.5.1 Cloud User Account and Password Policy 22
5.5.2 Data Classification and Encryption Policies 23
5.5.3 System Security Policy 25
5.5.4 Cloud Network Security Policy 25

6 Tools Building and Deployment 26

6.1 Cent OS and RHEL Hardening Tool 26
6.2 Cent OS Security Patch Checking Tool 26
6.3 Network Security Scanning 27
6.4 AWSSec Python Tool 29

6.5 AWSSec Windows Phone Tool 31
6.5.1 Managing Multiple AWS Accounts 32
6.5.2 Scan Configuration 34
6.5.3 Reporting 36
6.5.4 Deployment 45
6.6 Automated Audits by Cloud Services Team 45
7 Conclusion 46
8 Bibliography 47
APPENDIX 1: Hardening Script for Cent OS and RHEL 49
APPENDIX 2: Security Patch Checking Script for Cent OS 50
APPENDIX 3: List of Figures 52

Abbreviations / Acronyms

DevOps	A portmanteau of Development and Operations.
DC	Data Center
AWS	Amazon Web Services
VM	Virtual Machine
IaaS	Infrastructure as a Service
PaaS	Platform as a Service
SaaS	Software as a Service
AZ	Availability Zone
EC2	Elastic Compute Cloud
VPC	Virtual Private Cloud
EBS	Elastic Block Storage
AMI	Amazon Machine Image
S3	Simple Storage Service
RDBMS	Relational Database Management System
RHEL	Red Hat Enterprise Linux
CDN	Content Delivery Network

1 Introduction

To succeed in a competitive marketplace, organizations must be capable of rapid change, quick product releases, efficient communication, and low process overhead. Scaling up IT infrastructure such as servers and networking equipment is a capital intensive and time consuming process. Moving to cloud computing often has advantages of scalability, speed of execution and cost.

Software product development and services are also susceptible to similar market pressures. Organizations adopted agile principles to overcome some of the disadvantages of traditional software development methodologies. Agile methodology allows teams to embrace change, predictable delivery, and continuous stakeholder involvement.

DevOps is not a development methodology like Agile. It is a way of work that values multi-functional teams, collaboration, effective communication, automation and rapid releases. DevOps brings with it its own challenges, especially for corporate security assurance.

Security risks are amplified when the organization has massive cloud infrastructure utilized by dozens of discrete DevOps teams utilizing agile methods. Security governance and tools established for DC and waterfall development model cannot be used in such an organization.

This project explores the means of effective security governance in a corporation that is adopting AWS cloud, agile and DevOps at a massive scale. In particular, this research answers the following:

- What are the limitations of existing security governance processes and tools?
- Since automation is a key part of DevOps, what security processes can be automated?
- How does the organization manage security when teams are utilizing agile methods for rapid releases?

1.1 Method and Process

This research methods used were:

- Literature review [1] [2] [3] [4]
- Feasibility review of existing security governance tools and processes
- Customizing processes or tools, build or buy new ones.
- Architecture reviews with dozens of projects

Figure 1 illustrates the security governance process.

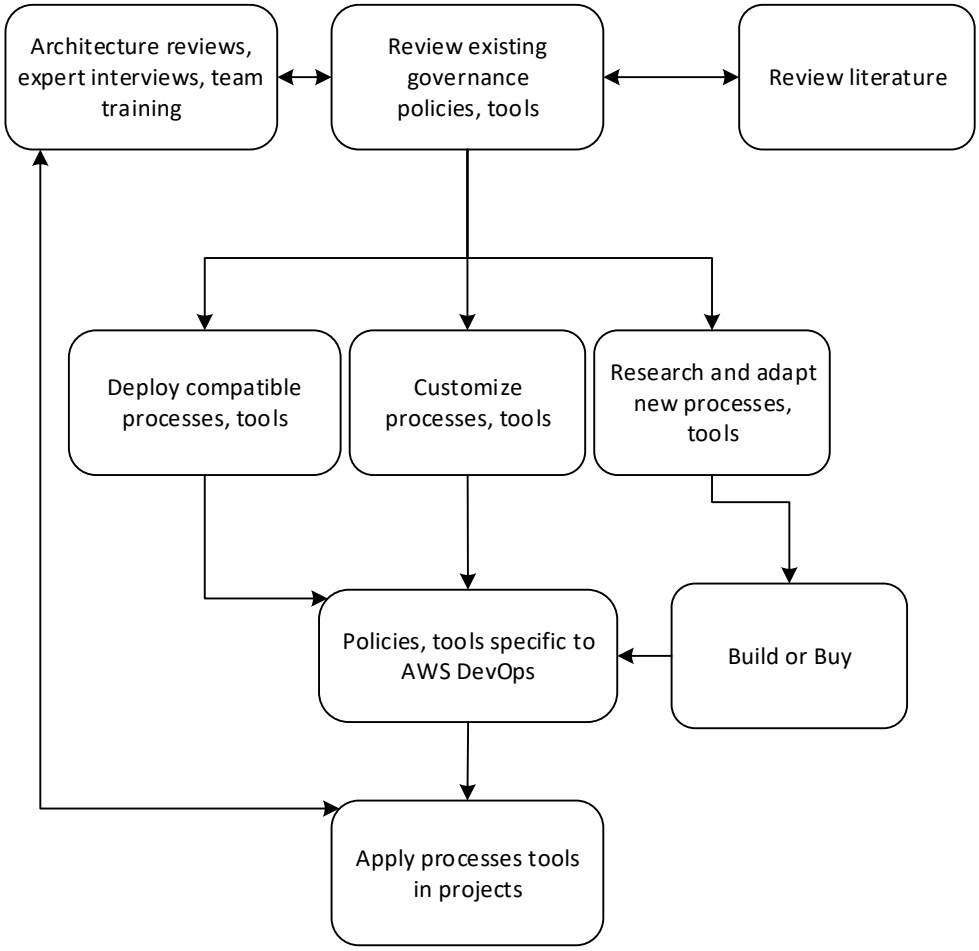

Figure 1: Security Governance Process

Risk management played a crucial role in determining which projects were prioritized for cloud deployment and thus be subjected to newer processes and tools.

1.2 Thesis Structure

Background information relevant to this research is presented in Sections 2, 3 and 4. Section 2 introduces some of the core domains of an information security governance program. Section 3 describes cloud computing in general, and AWS in particular. Some common cloud computing services provided by AWS is also described. Section 4 introduces agile and DevOps.

Sections 5 and 6 present the results of the research. Security governance and process results are described in Section 5. Tool building and deployment results are described in Section 6.

2 Information Security Governance

Information security governance consists of organizational structures, processes, policies for information risk management. [5] The key components of information security governance are [6]

1. Strategy
2. Roles and Responsibilities
3. Enterprise Architecture
4. Policies & Guidance
5. Implementation

This section describes the existing security governance components in the organization and their limitations.

2.1 Security Strategy

Organizations respond to customer, stakeholder, and thus business needs. The business strategy must be optimized to deliver maximum value to customers and stakeholders. IT outcomes required by the business strategy lead to the development of IT and Security Strategy.

The key goal of security strategy is to mitigate business and IT risks to an acceptable minimum level. Security policies, procedures and guidance that are well suited for traditional software development practices may need to be updated for DevOps and cloud.

2.2 Roles and Responsibilities

The security processes mandated by information security governance will also include the roles and responsibilities within the organizational structure. An organization may have a Network Operations team that is responsible for operations and security of networks within the organization's DC.

Traditional environments maintain separation of development, testing and operations personnel. This separation may not be well defined in a DevOps environment. Though

DevOps projects can contain roles that are primarily operations or development, it is fair to assume that developers can take testing or operational roles when needed.

2.3 Enterprise Architecture

In the absence of a coherent enterprise architecture, different units within the organization will develop IT solutions according to their own immediate needs. Enterprise Architecture teams can assist business strategy by providing seamless technology and process across the enterprise. This results in code reuse, unified technology and process stack and cost effectiveness.

2.4 Policies and Guidance

A security policy outlines specific information security objectives and the high-level strategy for securing data and assets. The security policy is composed by the security management and executive management. Guidelines provide recommended solutions to specific problems.

An organization may have a set of security policies, each targeting a different need. The policies are briefly introduced below.

Password Policy

User accounts must be protected against weak passwords. A password policy lays the foundation for securing and managing passwords across the organization. It also defines separate administrative and end-user responsibilities. Some of the common requirements enforced by this policy are:

- Password age
- Complexity
- Prohibition of reuse
- Prohibition of sharing
- Default password (randomness) and secure communication.

Sometimes the password requirements can be included in the User Account Policy which may include additional security requirements such as:

- Limit concurrent logins
- Lockout rules, such as conditions and length of lockouts.
- Provide provision for system (non-human) accounts
- Multi factor authentication

Privacy Policy

The Privacy Policy describes employee or customer data collection, uses, processing and storage. It must differentiate between private (to the employee or customer) data and other types of data (See Data Classification Policy). This policy must also address any applicable laws and compliance requirements related to handling of private data.

Data Classification Policy

Data that is collected, processed and stored by the organization can be classified and processed according to their sensitivity. It provides the foundations for protecting private and confidential data. Mandatory minimum level of access controls is defined for each classification.

Encryption Policy

The Encryption Policy provides information on approved crypto algorithms, key lengths and key management. It defines encryption requirements according to data classification. Some of the key policy requirements are:

- Approved algorithms and key lengths
- Key management – creation, storage, update and retrieval requirements.
- Encryption requirements for different storage (DB, laptop, OS drive) and communication mediums.
- Use of digital signatures

Network Security Policy

This policy aims to protect network assets and its users.

- Access control for wired and wireless networks
- Remote access by employees, contractors and third parties (such as suppliers)
- External network access (such as by customers)
- DC to corporate network access
- Security configuration and monitoring
- Secure disposal of assets
- User and administrative responsibilities

Organizations may contain two versions of this policy. One of them is the policy for protecting the office networks while the other aims to secure the data center.

3 Amazon Web Services Cloud Computing

This section provides a background on AWS, and describes the key components of AWS used in the company.

Section 3.3 describes a simple distributed architecture that will be used in this document to present the research.

Cloud computing refers to shared computing resources on a pay-as-you-go model available over the internet. Amazon AWS, Microsoft Azure and Google Cloud Platform are popular public cloud providers whose services are available to the public. Private cloud, on the other hand, is operated just for one organization. When an organization's cloud infrastructure consists of both public and private clouds, it is called a hybrid cloud.

Some of the key characteristics of cloud computing are [7] [8] [9]:

- Scalability – increase or decrease computing resources quickly to meet demand
- Metered service – pay according to usage
- Self-service over internet – ability to provision, manage and remove computing resources on-demand by self-service over the internet
- Resource pooling – the cloud provider pools its computing resources and serves multiple customers. Private cloud serves only one customer, but other advantages remain.
- Optimal resource utilization – computing resources are pooled and shared across different customers and applications.

3.1 Cloud Service Models

Cloud providers offer different services that utilize their cloud infrastructure [7] [8] .

Infrastructure as a Service (IaaS)

IaaS provides fundamental computing resources such as servers (virtual machines), storage (virtual disks, object storage), network (virtual subnets, software load balancers), etc. The customer has flexibility in designing their cloud infrastructure. AWS EC2 are virtual machines that are available as IaaS.

Platform as a Service (PaaS)

This service allows the customer to spend effort only on the application while the cloud provider completely handles the underlying cloud infrastructure. PaaS requires that the customer application is developed using a programming language and library supported by the provider. AWS Elastic Beanstalk is a PaaS.

Software as a Service (SaaS)

Cloud based applications that are owned and operated either by cloud providers or by third parties are called SaaS. Notable examples are Salesforce, Office 365 and Google Docs.

Amazon Web Services (AWS) IaaS and PaaS were used in most projects.

3.2 Amazon Web Services

AWS is a public cloud computing provider. It offers multiple cloud computing services under IaaS, PaaS or SaaS models.

Regions and Availability Zones

AWS datacenters are located across the globe. AWS Regions are geographic areas where there is AWS presence. Each region may have multiple DCs within. These are called Availability Zones (AZ).

Figure 2 illustrates the AWS regions and availability zones. Data transfers within AZ are free, while data transfers are charged for inter-AZ and inter-region traffic. Log servers are deployed intra-AZ and only the aggregate data is transferred out to other AZs.

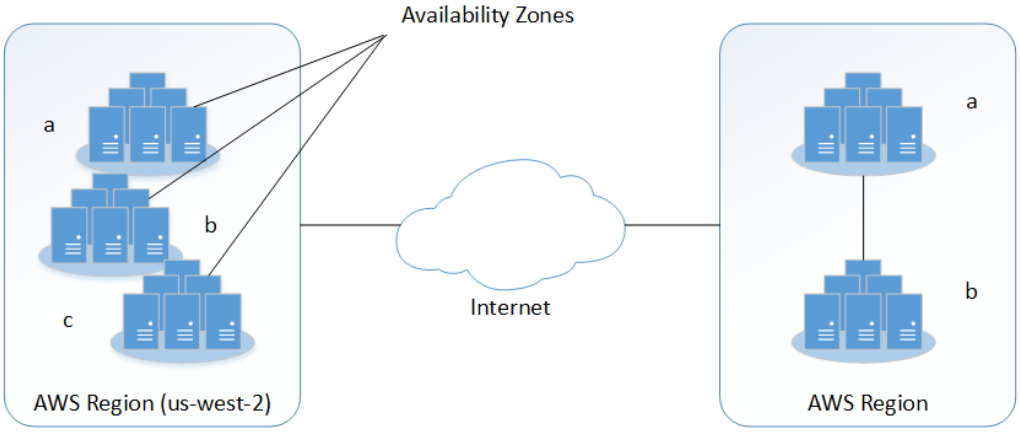

Figure 2. AWS Regions and Availability Zones

The web services served millions of users worldwide. To serve this geographically distributed customer base, and for performance and availability reasons, many of the products were architected to utilize multiple AWS regions and availability zones.

Elastic Compute Cloud

Elastic Compute Cloud (EC2) service provides virtual machines that the customers can rent. Customers can select from different instance types according to their work-load. Amazon Machine Images (AMI) are read only file systems that fare used to create a virtual machine instance. AMIs are available for different operating systems and versions.

Virtual Private Cloud

Virtual Private Cloud (VPC) is a virtual network environment where cloud resources can be isolated. Customers can choose their own IP and subnets, configure routing and gateways.

Simple Storage Service

Simple Storage Service (S3) is an object storage service. The maximum size of an object is 5 terabytes and it can have up to 2 kilobytes of metadata. Objects are stored into virtual containers called buckets. Both the buckets and objects are accessible through HTTP interface.

Elastic Block Storage

Elastic Block Store (EBS) are persistent block storage for EC2 instances. EBS can also be used as a root volume for EC2 instances. EBS backed EC2 instances will retain their data after a shutdown or termination.

EBS volumes can be attached to EC2 instances to provide additional storage space.

Relational Database Service

Relational Database Service (RDS) provides managed relational databases as a service. Customers can choose from MySQL, PostgreSQL, MariaDB, Oracle, Microsoft SQL Server and Amazon Aurora. Amazon takes care of the DB administrative tasks thus freeing the customer to concentrate on the application development.

Identity and Access Management

Identity and Access Management (IAM) provides user management and access control for AWS resources. Users can be given permissions for accessing specific AWS resources that they require.

3.3 Sample Web Service Architecture in AWS

The systems in use serve millions of customers across the globe. Projects use geographically distributed cloud architectures for availability and performance. A simple distributed architecture is given below (Figure 3).

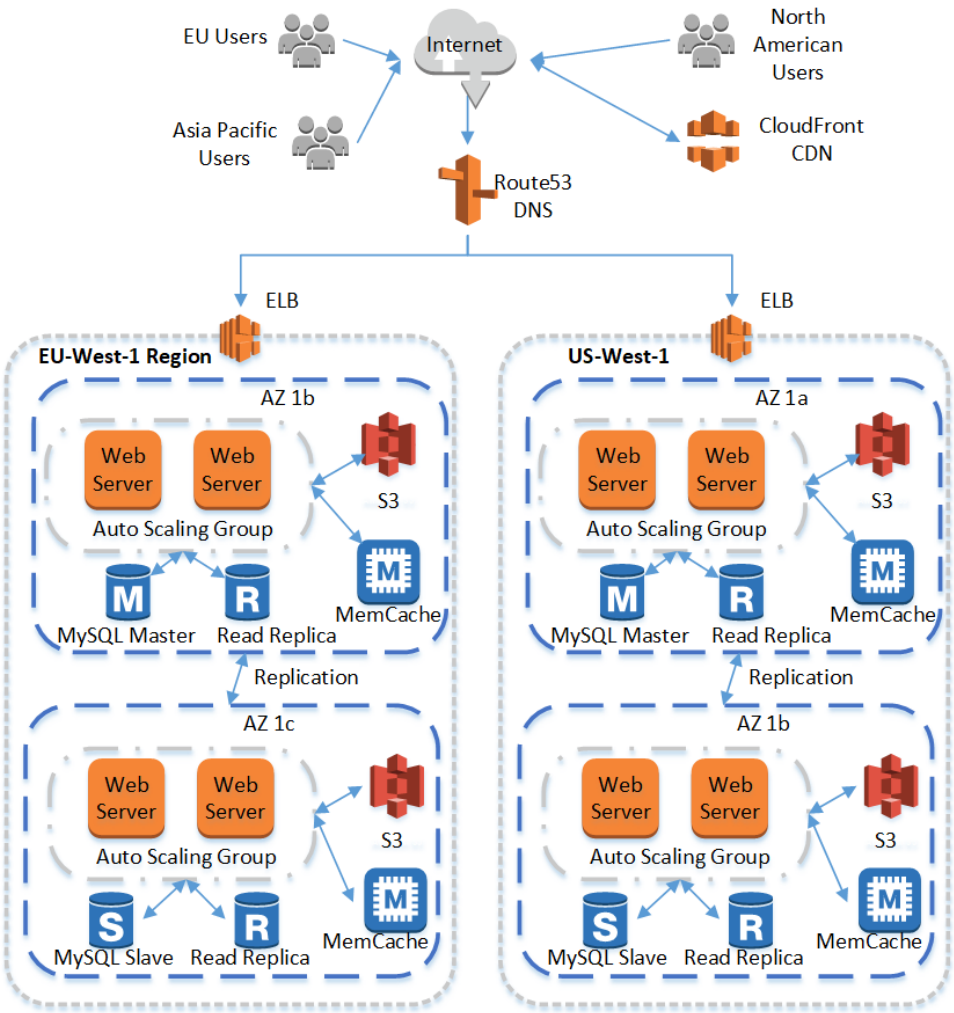

Figure 3: Sample Web Service Architecture in AWS

Since most users are geographically distributed, multiple AWS regions are used by projects. When a customer accesses the service, Route53 will route the request to the region nearest to the customer. This distributes the load and provides better performance by reducing the RTT (Round Trip Time).

Within each region, the projects are recommended to distribute their computing resources in more than one AZ. In the architecture shown above, each region has one

master, one slave and one read replica DB. The master and slave are located in different AZ.

Web UI and business logic may be served from EC2 instances that are auto scaled. During sustained load periods such as in holiday season, each AZ may have hundreds of auto scaled instances. The number of instances can vary significantly by the hour.

Static objects such as photos and videos uploaded by the customers will be stored in S3 buckets. Some of these public objects may be pushed to Cloud Front CDN for higher performance.

Distinction must be made between such publicly accessible S3 objects and other data such as service configuration data objects. These sensitive data are required by the web service itself and must never be exposed to the public. The project may use different S3 buckets for such segregation.

4 Agile and DevOps

This section describes the software development methodology and practices used in the organization. The impact of these practices on security governance is also discussed.

4.1 Agile Methodology

Agile development is a way of thinking about software development. It focuses on building software in small iterations so that working code is delivered and tested frequently. The iterative releases also enable efficient adaption of changes in customer requirements. The Agile Manifesto describes a collection of 4 values as the canonical description of this way of thinking [10].

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Agile development is better suited for responding to changes than waterfall model. Agile methods promote adaptation and small frequent releases. Frequent releases enable the end user to test the small changes, and this in turn provides valuable feedback to the development team.

There are different approaches to agile development [10]. Some examples are:

- Scrum
- Kanban
- Extreme Programming (XP)

Scrum is commonly used within the organization in question.

4.2 DevOps

Unlike Agile methodology, DevOps is a way of work that emphasizes on multifunctional teams and prioritizes quick releases of working code. In this approach, development, test and operations teams are merged into a single entity for effective collaboration and

rapid releases. It is typical for engineers to have multi-functional roles in smaller and less complex projects. Large or complex projects will usually some dedicated specialists for test or operations, but they will still be part of a single DevOps team. Figure 4 shows the DevOps process.

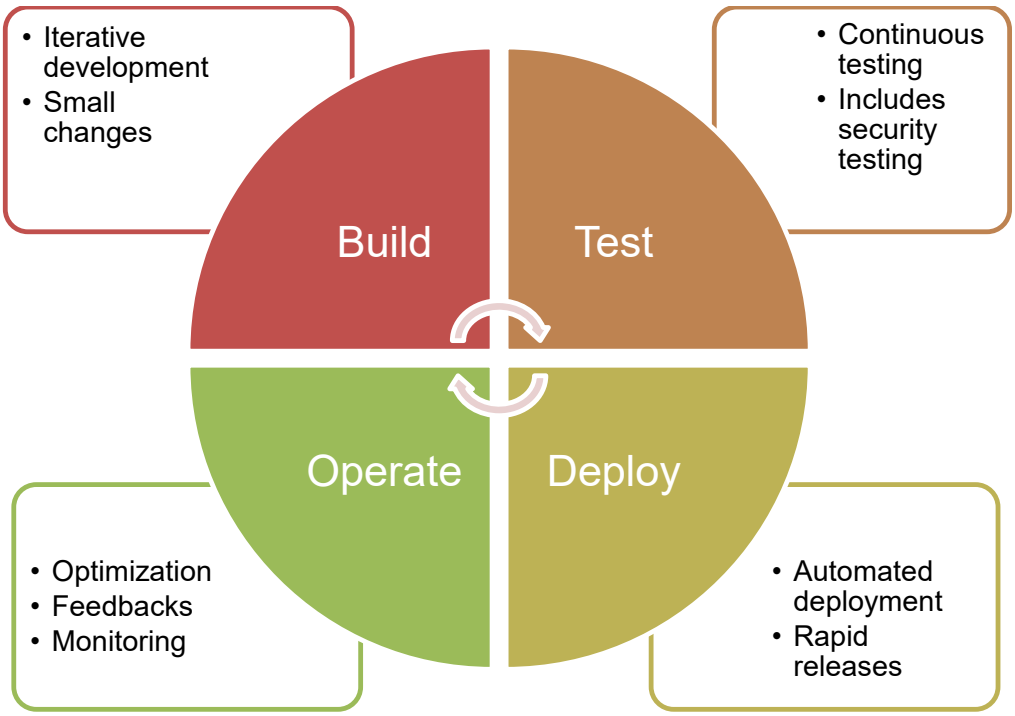

Figure 4. DevOps

DevOps requires organizational changes to support multi-functional teams, automation and continuous delivery. Agile and DevOps share many common goals.

4.3 Security Governance Must Be Agile

Many existing security governance processes such as operations security controls were ill suited for cloud deployments and DevOps practices. The mandate for the security team was that the processes must be updated to address the new development practices.

While DevOps enabled faster time to market, it also introduced new risks. Experienced operations team were administering the own data centers and the existing security processes were designed to prevent any untoward exposure of the services.

Exposing a service or server to internet was only possible after clearing multiple security controls. But any developer in a DevOps team could bring up a server and expose it to internet in mere minutes without any security oversight.

5 Security Governance and Process Assessment

This section describes the security governance and process changes the organization in question instituted to adopt to the cloud DevOps environment.

The organization has legacy and new projects. The projects use diverse technology stacks. AWS is currently the primary cloud provider, but there are some Azure and private cloud deployments. This research focuses on AWS deployments.

Individual projects in the company can follow different agile methodologies and DevOps. Some of them release minor changes to their products and do so frequently. Others follow a more traditional approach where their release cycles are longer and bigger. An enterprise level security team is responsible for security of all projects in the company.

The security team, of which the author of this thesis was part of, was responsible for the cloud security governance and process assessment. The security team evaluated and adopted the following governance and process changes. A number of Amazon Web Services security literature were reviewed in the process. [11] [12] [13] [14]

5.1 Security Strategy

This section describes the strategy for AWS cloud adoption in a secure manner. Cloud adoption brings new security challenges. Data is stored on the internet, in cloud provider data centers. Moving to the public cloud can mean some of the security controls present in customer's own data centers are unavailable.

The strategy was to adopt AWS in stages based on risk assessment:

- Encourage new projects to architect directly for the cloud and deploy in it.
- Small projects that are less business critical, and those that required no significant changes for AWS, were selected for adoption.
- Some legacy products relied on system configurations unavailable in AWS (such as large memory Oracle DB servers). Such projects can continue to use the data centers till such time when they were re-architected for AWS.

- Risk assessment also considered the sensitivity of the data. Projects that held secret or highly sensitive personally identifiable information were not targeted for AWS adoption.

Figure 5 illustrates the cloud adoption strategy at hand.

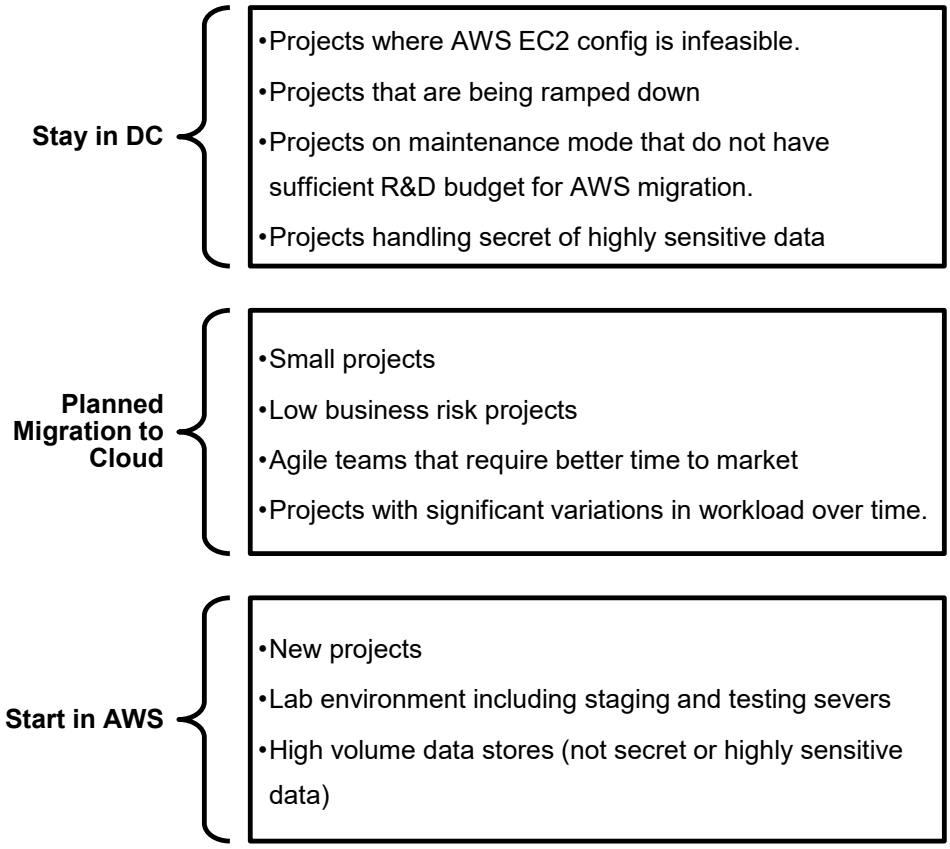

Figure 5: Cloud Adoption Strategy

New projects needed very little effort in cloud adoption. Early adopters were architected specifically using plain IaaS components and deployed exclusively to the cloud.

5.2 Adopting Security Processes to DevOps and Scrum

In scrum and DevOps, the software development phases (called Sprint) can be short and rapid. Three stages of product development were identified where the security team must be engaged. [15]

The main change to the security processes was to train the DevOps teams on security so that they can self-certify each scrum. Since the DevOps team may not have security experts, and since each scrum execution is only a few days long, automated tools were used for self-certification. Automated security testing was introduced in scrum along with other testing phases like User Acceptance and Load Testing [16]. Components of a secure scrum are illustrated in Figure 6.

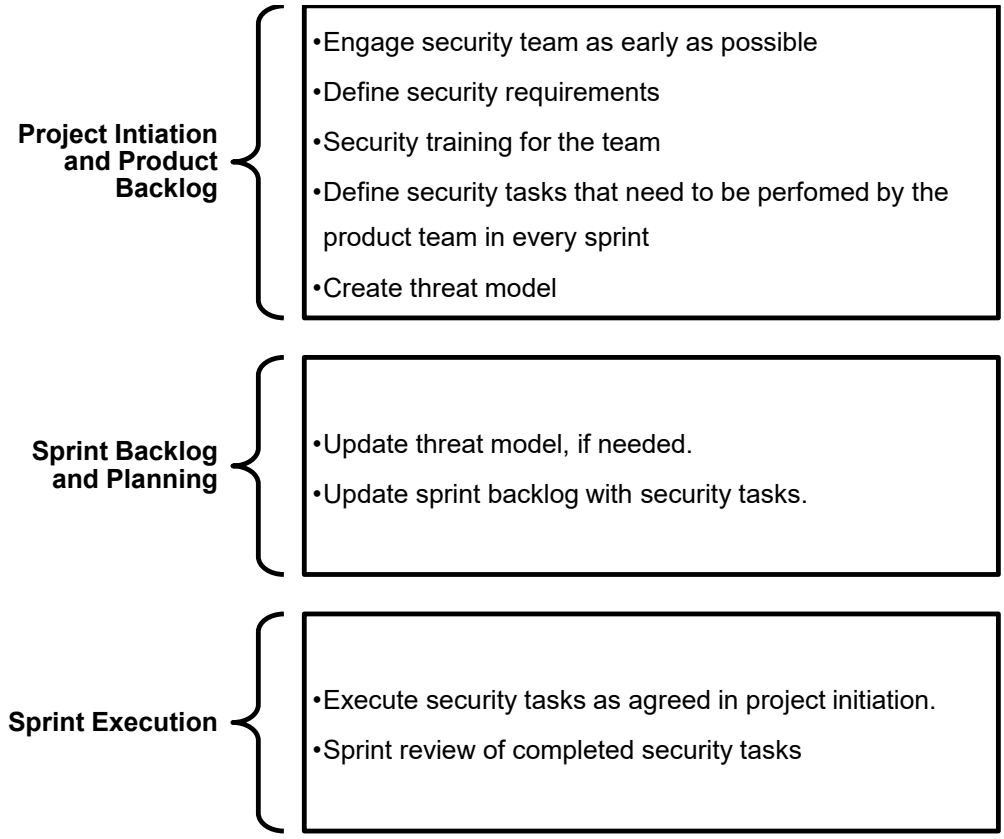

Figure 6: Secure Scrum

An exception for self-certification was provided when significant architectural changes were done in an iteration. On such changes, the security team must be engaged and the threat model will be updated as needed.

The diagram below (Figure 7) describes the security tasks to be carried out by the development team during each sprint.

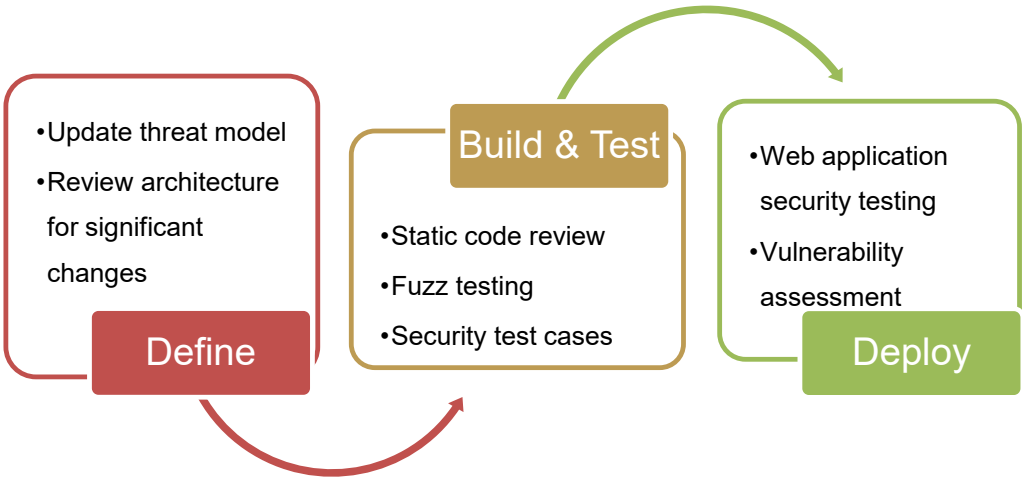

Figure 7. Security tasks for every sprint

Static code reviews and security test cases are automatically run on every build. Web application vulnerability scans are scheduled to run periodically in testing servers.

5.3 Organizational Structure and Responsibilities

A new team was formed that will provision AWS subscriptions for other projects. The cloud services team was responsible for various common cloud operations and security activities. Many of these tasks will be described later in this thesis.

The cloud services team had the “master credentials” with which they had access to all AWS resources within the company. Only the cloud services team can create new AWS

subscriptions and projects were prohibited from creating their own accounts using corporate credit cards. Each project received two AWS subscriptions – one for production and one specifically for R&D.

5.4 Production and R&D Accounts

R&D environments posed a higher security risk due their nature. While the production environments are changed less often, R&D environments will undergo rapid and frequent changes.

The production environment can contain sensitive data and code. This environment requires stricter security controls and more security effort than R&D environments. Segregating R&D environments inside their own VPCs but within the same AWS account as production had its own challenges. AWS VPCs were common in production to segregate different components of the service. When R&D environment is in the same AWS account, this adds a number of VPCs. Tracking VPCs according to their production or R&D affiliation and ensuring that no production data can ever be copied to R&D was a too great configuration management challenge.

Creating separate AWS accounts for R&D and production simplified configuration management and security governance. Figure 8 illustrates this change.

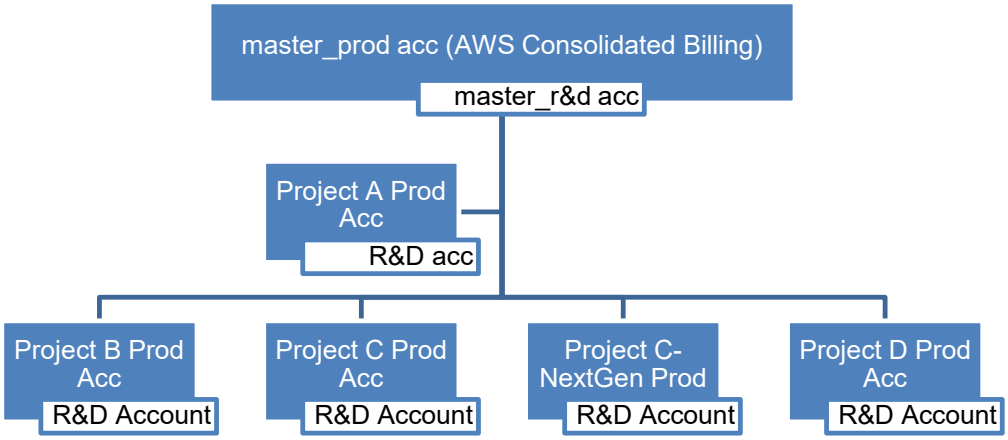

Figure 8: Cloud Account Management within the Organization

Thus all projects were provided with separate production and R&D AWS accounts. These two accounts were identical except for a consistent naming. Data transfer from production to R&D accounts was prohibited by policy.

5.5 Policies and Guidance for AWS Projects

This section describes the changes to security policies and guidance that were adopted for the managing cloud DevOps security.

5.5.1 Cloud User Account and Password Policy

Projects will create AWS IAM user accounts for team members who need to access AWS resources. Projects also needed to create system accounts in AWS IAM. These system accounts were not identified with a human user, but were used to run batch jobs and automation. AWS doesn't distinguish between user and system accounts within IAM.

Some of the key user account policy changes proposed by the cloud services team were:

- All IAM user accounts must be named after the user's corporate Active Directory User ID.
- AWS administrators within the project's AWS account must use their own user ID. This user ID must belong to Administrators group.
- All IAM system accounts must be named in a consistent format as defined in the policy.
- Password rules were similar to corporate Active Directory passwords. However, all IAM user accounts must have two factor authentications enabled.
- Cloud services team ran periodic audits to on all AWS IAM accounts in the organization to verify compliance to this policy.

5.5.2 Data Classification and Encryption Policies

All data stored in AWS must be classified according to organization's data classification policy. While the data classification policy was unchanged, a new Encryption Policy was used to define which classification levels must be encrypted during storage and transfer. Figure 9 shows the data classification.

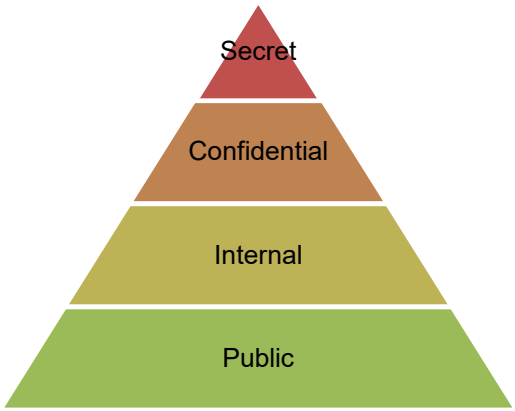

Figure 9: Data Classification

Encryption is mandatory for all confidential and secret data stored in AWS. Access keys for servers and web services are considered secret data.

The encryption policy provided encryption requirements and guidance for different AWS components. This policy addressed encryption requirements for major AWS components

that were used by projects. The table given below (Figure 10) shows some key encryption policy requirements.

Instance (AMI) Storage	•Do not store cleartext secret data in instance storage. •Do not store encryption keys in instance storage. Instance storage is destroyed when instance crashes, so encrypted secrets must always be backed up.
EBS	•The AMIs provided by the security team do not have encrypted root device. •Use Cryptsetup to create encrypted partitions in new EBS. The passphrase must be securely stored. •The EC2 instance will need the passphrase at boot time to mount the encrypted volume. It is permitted to store disk encryption keys in config file such as Cloud Formation or Beanstalk launch configuration.
S3	•Sensitive data in S3 must be stored in encrypted form. •S3 supports server side encryption which is transparent to the application code. •Proper key management procedures must be implemented if the project utilizes its own encryption and decryption routines. This requirement also applies to S3 Client Side Encryption.
RDBMS in IaaS Mode	•Projects utilizing RDBMS such as MySQL in IaaS mode can utilize encrypted EBS volumes for DB data storage. •Some RDBMS such as Oracle or SQL Server support transparent database encryption. This may be utilitized when the underlying EBS volume is not encrypted.

Figure 10: AWS Encryption Policy

The encryption policy described the technical security controls required for different AWS components. It also provided clear examples for exceptions for encryption.

5.5.3 System Security Policy

The server operating system used in production environments, and to some extent in R&D environments must be hardened. The cloud services and security teams were responsible for creating secure hardened AMI for use by the projects.

The hardened AMI addressed the security configuration of the following OS areas:

- Password based logins for Linux EC2 instances are disabled. SSH key based logins must be used.
- Direct root login is prohibited. Sudo must be used.
- Keys must be periodically changes.
- No root login is permitted.
- Enable central logging of important system events.
- Automated log monitoring must be done.

The author of this thesis developed the initial version of the hardening and patch checking script which was taken into use and maintained by the security team of the organization.

5.5.4 Cloud Network Security Policy

It was established that limiting administrative access of AWS resources to the corporate network provided enhanced protection. All projects were required to configure AWS Security Groups such that

- Administrative access (SSH, Remote Desktop) to be allowed only from specific source IP ranges only.
- EC2 and data storage cloud services must restrict open ports
- VPC security groups must enforce strict inbound and outbound security rules.

6 Tools Building and Deployment

Automation is a key feature of enabling DevOps. Core security tasks were evaluated and automated to fit into the project's agile lifecycle.

Some of these security tasks were delegated to the teams themselves. Sometimes, the automated tools had to be simplified so that the project teams can use them directly.

6.1 Cent OS and RHEL Hardening Tool

A script to validate OS hardening for Red Hat Enterprise Linux and Cent OS 5 and 6 versions. The script was written in Perl by this researcher and validated several recommendations from the RedHat Enterprise Linux 6 Security Guide [17] . The code snippet provided below shows the different hardening sections (available_audits variable) that the script validates.

The script was used during the creation of the hardened AMI to ensure secure OS configuration. The project teams used the script during sprint validations to ensure that the current sprint did not inadvertently roll out insecure configurations.

Code snippet for the Perl hardening script is provided in Appendix 1.

6.2 Cent OS Security Patch Checking Tool

Another key system security requirement was that the EC2 instances must regularly apply security patches as needed. Red Hat Enterprise Linux provides yum-security plugin which enables the system administrators to search and install only the security patches. CentOS is a Red Hat Enterprise Linux compatible free distribution. At the time of this research (circa 2014), yum-security plugin did not work correctly in CentOS which made it difficult for the operations team to identify and prioritize security patches.

A patch checking script was developed in python to enable the operations team to list unapplied security patches in a given Cent OS installation. The script obtained security patch information by parsing the Cent OS announce mailing lists (HTTP) for security

patch releases. It parsed updated package name and version information from the mailing list announcements and created a flat file DB of the results. This flat file DB was then used to assess the patch status of installed packages.

The DevOps teams can schedule this script in production instances to audit and log security patching status.

Code snippet for the python patch checking script is provided in Appendix 2.

6.3 Network Security Scanning

Qualys and Nmap network security scanning tools were used in the data centers. A separate task force was formed to adopt and deploy these tools in the AWS infrastructure.

Qualys' AWS scanner is called the Virtual Scanner Appliance. These scanners need network connectivity to their target EC2 instances. Projects have to deploy multiple Virtual Scanner Appliances such that every subnet within every VPC is scanned. When auto-scaling is used, AWS will launch many new instances based on demand. Subnets with potentially hundreds of auto-scaled instances were granted an exception to scanning because that will impact the resource consumption. Instead, a copy of these instances was scanned in staging environment.

Nmap is an open source network mapping tool. It was used to scan the publicly accessible systems for open TCP and UDP ports. Such security scans must be done only after obtaining permission from AWS. AWS provides the permission and whitelists the source IP address(es) from which the scans will originate. Since the source IP of the scanning server must be static, public IP addresses were assigned to the scanning server.

Public IP addresses that were not owned by the researcher's company were required for the scanning server. The AWS security group rules allowed extra open ports such as administrative access (port 22/SSH) when the source IP address was within known IP address list. If the scanning server was located within the corporate network, the AWS security groups will allow incoming packets to administrative ports as well. Public IPs that were not listed in the corporate IP list enabled the scanner server to produce a more

true result of publicly visible AWS systems. Figure 11 illustrates the network scanning deployment.

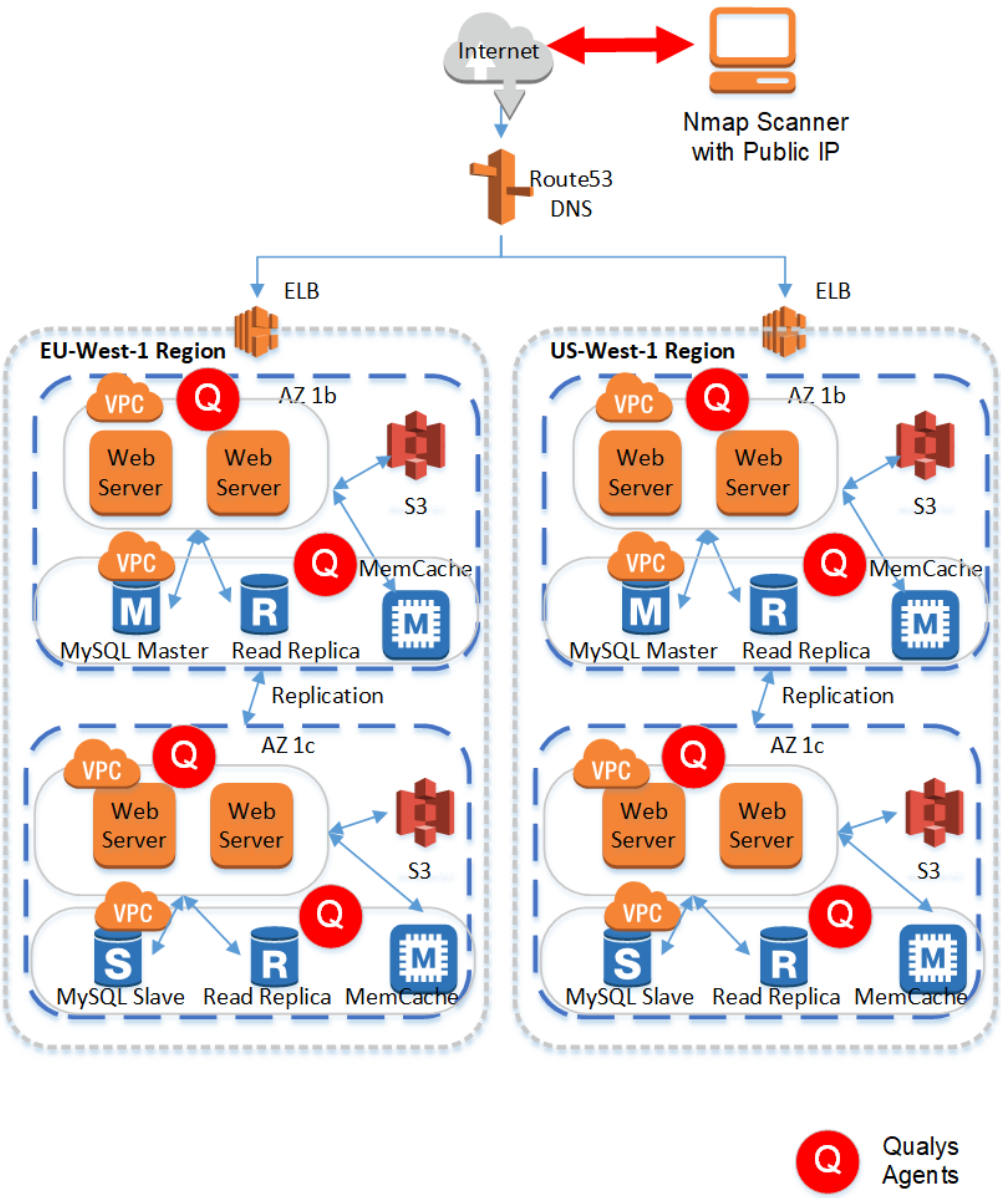

Figure 11: Network Scanning Deployment

The above figure illustrates the deployment of Qualys scanners within each VPC. VPC must have appropriate security group rules configured for Qualys. Qualys deployment was centrally managed. The scanner appliances can be deployed by the project teams themselves.

6.4 AWSSec Python Tool

DevOps teams needed security tools to perform self-assessment. The existing network scanning tools were unsuitable for direct use by the DevOps teams because:

- Nmap scan of a large account with thousands of auto scaled EC2 instances can take days.
- Qualys scans are centrally managed and DevOps teams neither had the Qualys expertise nor the time to setup and maintain it in their R&D accounts.

Every project had two AWS subscriptions: one for production and another for development and staging. The production subscription will expose project resources such as web servers or public s3 objects and this is well documented. The development subscription must not expose any resources to public. The requirement was to build a tool that:

- Stores a whitelist of publicly accessible resource URIs for each subscription (AWS account)
- Rapidly scans multiple accounts and notifies of violations.

A python tool was created by the author of this thesis for this requirement. It utilized AWS API to query all instances and security groups within an account. It then correlates this information and lists all the instances with ports open on the internet. The python script produced a simple textual output.

The python tool can complete a scan of thousands of EC2 resources in a few seconds. Figure 12 shows the scan results of AWSSec Python tool.

```
Getting all instances... done. Got 1003 instances.
Getting all security groups... done. Got 20 security groups.
-------------------------------
i-afc49d9c default 54.[redacted]140  running
[192.100.104.0/21] 80
[192.100.104.0/21] 443
[None-683904650913] None
[None-683904650913] None
[None-683904650913] None
-------------------------------
i-15fea726 default 54.[redacted]145  running
[192.100.104.0/21] 80
[192.100.104.0/21] 443
[None-683904650913] None
[None-683904650913] None
[None-683904650913] None
-------------------------------
i-b70e2984 default 54.[redacted]144  running
[192.100.104.0/21] 80
[192.100.104.0/21] 443
[None-683904650913] None
[None-683904650913] None
[None-683904650913] None
-------------------------------
i-77406a44 default 54.[redacted]  running
[192.100.104.0/21] 80
[192.100.104.0/21] 443
[None-683904650913] None
[None-683904650913] None
[None-683904650913] None
-------------------------------
i-77406a44 Itasca-SSH 54.[redacted]  running
[199.177.12.0/24] 22
-------------------------------
i-a3406a90 default 54[redacted]  running
[192.100.104.0/21] 80
[192.100.104.0/21] 443
[None-683904650913] None
```

Figure 12: Python AWSSec output showing a violation

AWSSec Python tool produced textual results and the red highlight was added by this researcher to point out a violation in the report. Sensitive data (IP) in Figure 12 has been masked by the author (researcher).

6.5 AWSSec Windows Phone Tool

The AWSSec Python tool required a non-corporate public IP address to provide results without false positives. This requirement was an additional effort that many small DevOps teams found unfeasible.

This drawback was eliminated by rewriting AWSSec as a Windows Phone app developed in C#. The AWSSec Windows Phone app can utilize the phone's mobile internet to perform the scans when wireless network was turned off. The phone's public IP will then be that of the mobile network carrier's IP range. The scan results did not show the extra ports open only to the corporate network.

Like the python tool, Windows Phone tool used AWS API to obtain the list of deployed AWS resources and their security groups. AWSSec then correlated this information with a whitelist of publicly accessible resources. It added a functionality to whitelist ports that the project team expects to be exposed to internet. When whitelisting is enabled, the tool will only report ports that are open but are not on the whitelist. Figure 13 shows a simple illustration of the core tasks performed by the AWSSec Windows Phone tool.

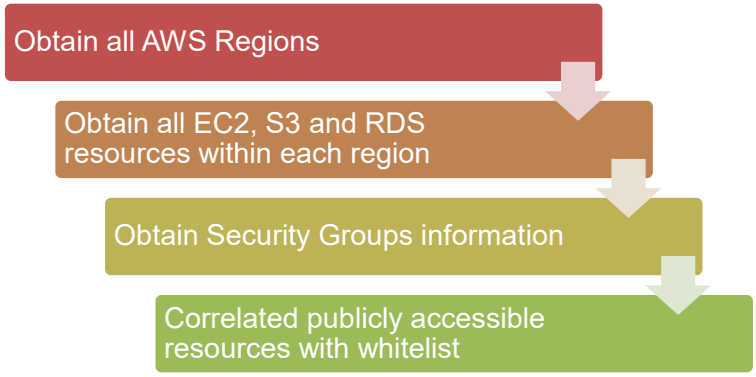

Figure 13: AWSSec Tool Tasks

While the core tasks are similar to the Python version, the AWSSec Windows Phone Tool contains additional features that assist projects in self-certification. Some of the key features of the tool are described below.

6.5.1 Managing Multiple AWS Accounts

AWSSec tool supported multiple AWS accounts. The screen capture provided below shows two configured AWS accounts. Please note that projects will typically their production accounts configured as well. Figure 14 shows the main screen of the AWSSec Windows Phone app with two configured accounts.

Figure 14: Multiple Accounts in AWSSec

The toolbar at the bottom provided commands for account management and settings.

Project teams typically added both their production and R&D accounts in the tool. New accounts are added using the “add account” page of the mobile app. For EC2 scanning, expected open ports must also be specified in this screen.

Figure 15 shows the “add account” screen which is launched by selecting the “+” icon in the main screen.

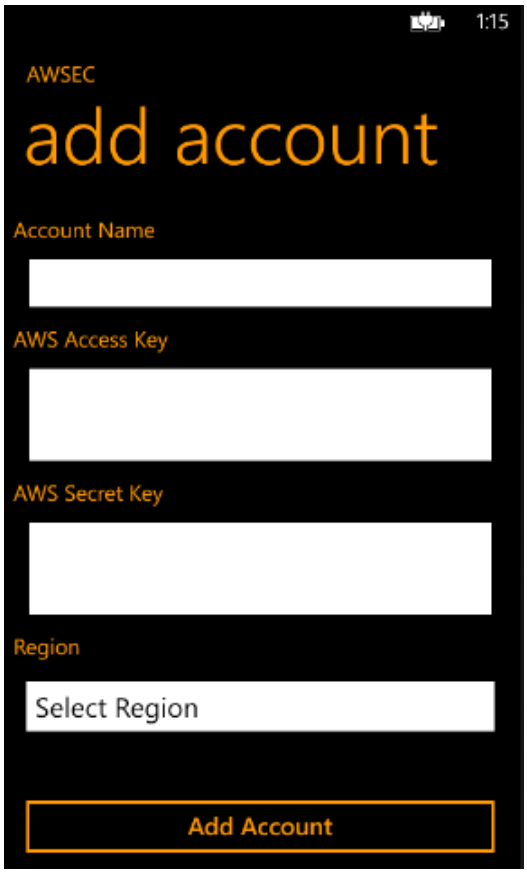

Figure 15: Add Account Screen of AWSSec Tool

Account name is a textual description. The tool scans all regions by default. AWSSec tool stores AWS Access and Secret keys in encrypted form using ProtectedData class of .NET Framework.

AWSSec tool can scan multiple accounts in parallel. Accounts can also be limited to specific regions, or scan all regions. AWS has a separate region for US Government projects. The option “All Regions” under “Select Region” will scan all regions except AWS Government. Figure 16 shows the list of Regions supported by the app.

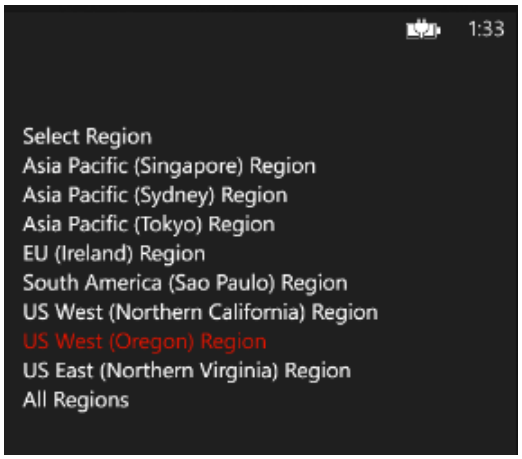

Figure 16: Scan Settings - Region Selection

A limitation of the current version of the tool is that it doesn't support selecting two or more regions specifically. The projects can select one region or choose all regions. This limitation was a minor usability bug because the projects almost always want to scan their entire AWS infrastructure.

6.5.2 Scan Configuration

Scan settings screen can be opened by selecting an account and pressing "Settings" button in the main screen.

Figure 17 shows the configuration options available in the scan settings screen.

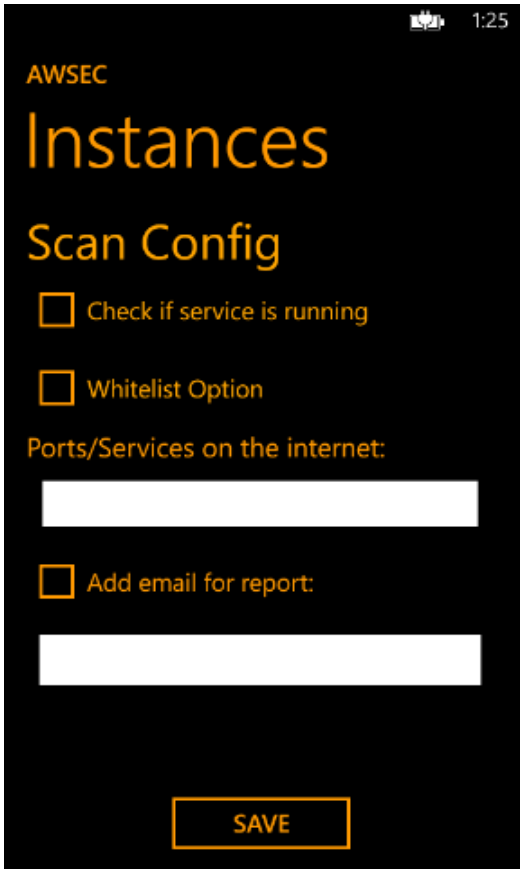

Figure 17: Scan Settings Screen

When “Check if service is running” option is selected, AWSScan will attempt to connect to open ports discovered through AWS API.

The “Ports/Services on the internet” text box accepts a comma separated list of ports. If the “Whitelist Option” checkbox is selected, the list of ports are considered to be acceptable open ports for this account. Whitelist examples are ports such as 80 and 443.

If the “Whitelist Option” checkbox is unselected, the list of ports are considered as a blacklist. AWSSec will report any such port findings regardless of whether “Check if service is running” returned true.

AWS accounts can have thousands of EC2 instances and millions of S3 objects. The scan report can be very large for viewing in the small phone screen. The email option can be used to email the report to user.

6.5.3 Reporting

Figure 18 shows a scan in progress.

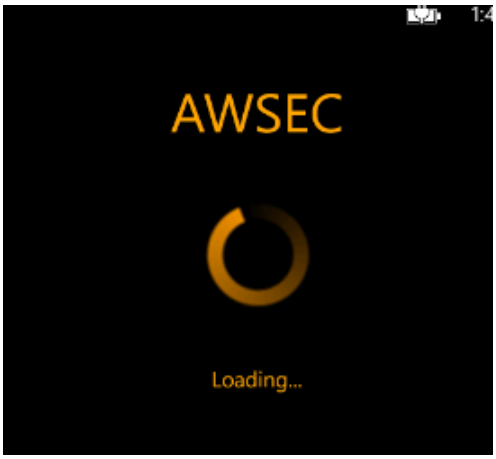

Figure 18: AWSSec Scan in Progress.

The scan progress screen merely displays an animation to indicate the scan is ongoing. There are no options to cancel a scan that has started. But the user can kill the phone app to cancel it. Most projects can complete the scan in under 15 seconds, so the lack of scanning control was not an important feature request.

Project Scan Showing No Violations

The “Instances” screen displays the scan results. When there no violations are found in the scan, this screen is empty and the “Send Email” button is disabled. Well managed projects and scan settings will always result in this screen. Figure 20 shows the scan results where there are no violations.

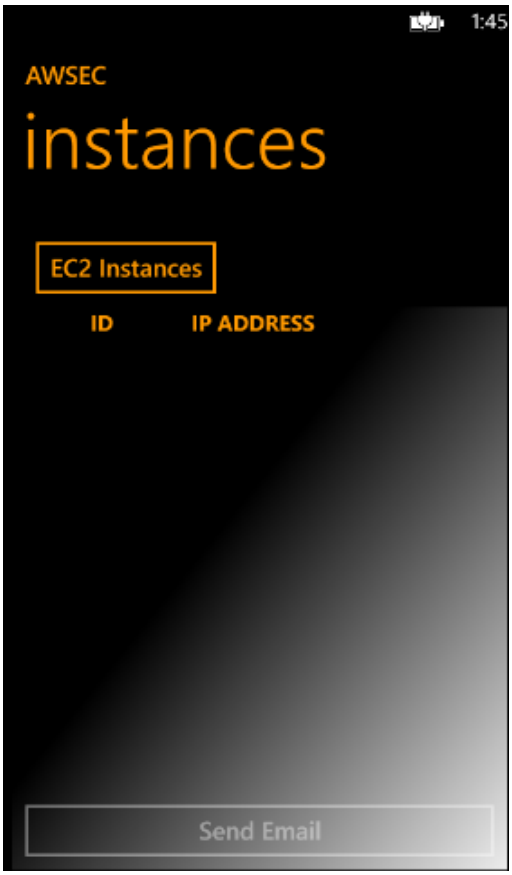

Figure 19: Account with no violations.

The “EC2 Instances” button will rerun the scan and populate the results in the same page if there are any findings.

Project Scan Report Showing Violating Instances List

If there are violations, the list of instances are shown. Figure 20 shows a scan where one instance had violations.

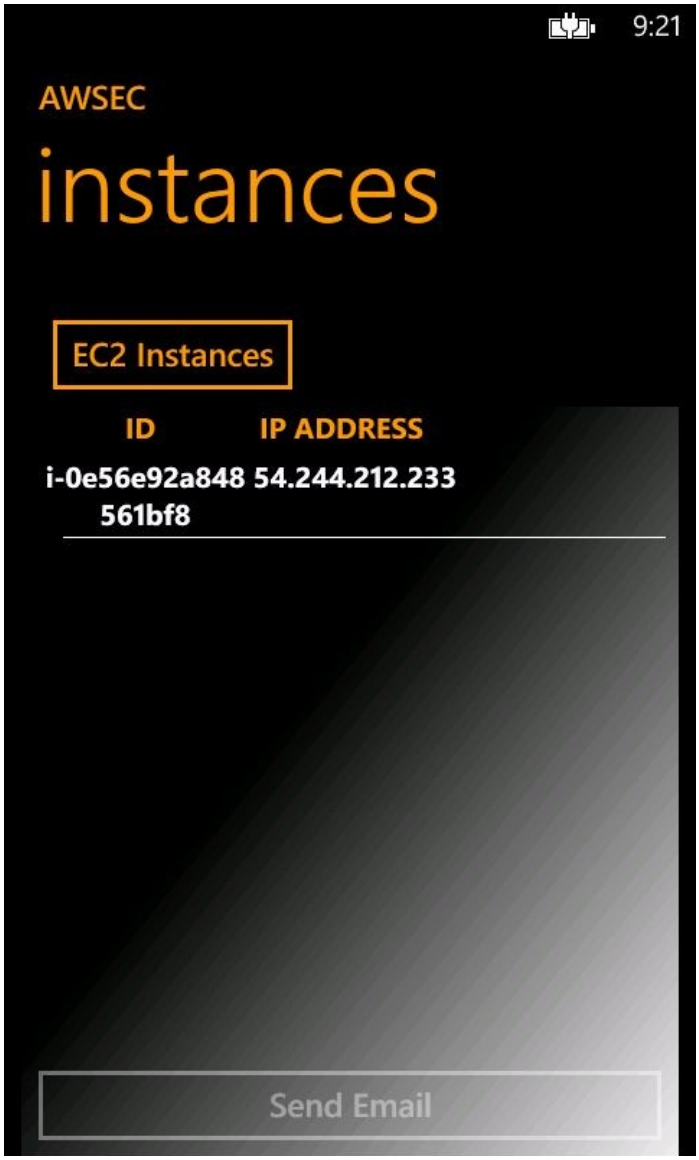

Figure 20: List of Scan Violations

The ID column displays the EC2 Instance ID and the IP Addresses column show the public IP of the violating instance.

If there are multiple violations the list is populated with one EC2 instance per row. Each row can be touched to obtain more information. Figure 21 shows a prototype screen with the list populated using scan violations.

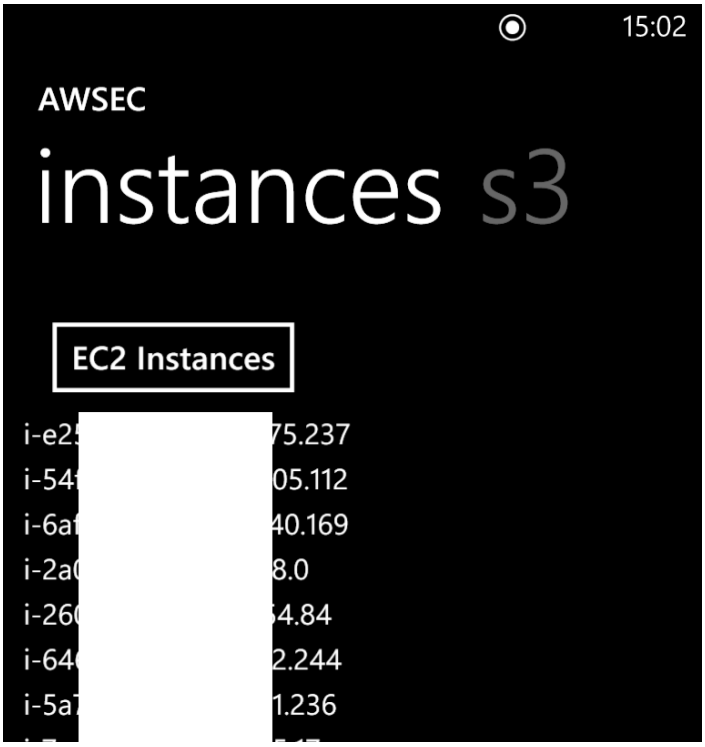

Figure 21: Scan report showing multiple violations (in prototype UI)

Sensitive data (public IP) has been masked by the researcher in the above Figure. If the list of violations is too long it can be difficult to review it in the mobile screen. The “Send Email” option can be used to send the report out to an email address.

Instance Violation Details

The user can review the report of each violating instance by touching on it. The tool provides three types of information on violating instances:

General Details screen provides basic instance metadata including its public IP and DNS name. Figure 22 shows the details of a instance with scan findings.

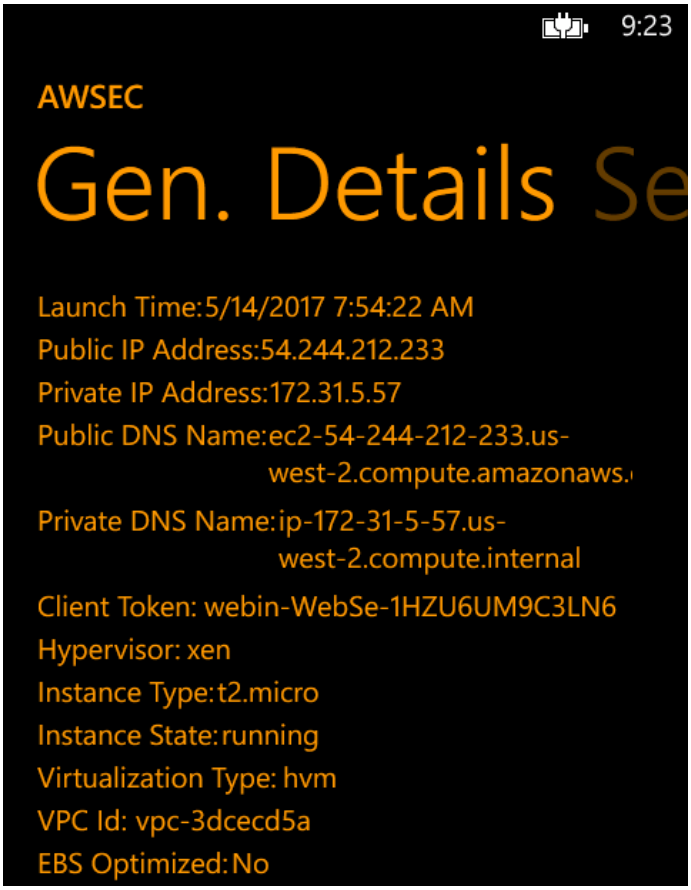

Figure 22: Instance Details in Violation Report

Apart from the public IP, the general details screen also shows launch time, private IP and DNS name.

The next tab called Security Groups shows the security groups that are protecting the instance. Every instance must be protected by a Corporate IP Security Group. This security group allows administrative access only from the corporate network and blocks it for others. Figure 23 shows that the instance is protected by one security group.

Figure 23: Not Applying Corp IP Security Group is a violation

The screenshot presented above shows that the corporate IP security group was not applied. This can result in administrative access ports such as SSH/22 being open to the internet.

The third tab is Ports and is described below.

Pinging Open Ports

By default, AWSSec tool uses AWS API exclusively for its scan. The third tab of the instance details shows the list of open ports obtained by querying AWS API for security group rules. It does not truly validate whether these ports are open in the underlying instances.

There are two cases when attempting to open a TCP connection to open ports is needed. The first case is that of the security group misconfiguration. One of the security group rules might have opened a port when the underlying service does not need it. The second case pertains to availability of services. The web service running in the instance might become unresponsive.

This default behavior can be changed in the scan configuration as described in section 6.5.2.

Warning: Enabling the “Check if service is running” option as described in that section will lead to significantly slower scan times in accounts with thousands

of EC2 instances. Windows Phones implement an automatic screen lock after a period of inactivity and that event will pause AWSScan app. Projects are thus advised to enable that feature only when needed. Figure 24 shows that ports 80 and 22 are open.

Figure 24: Ports tab showing two possibly open ports

The ? symbol next to open ports shows that these ports have not been “pinged”. Clicking on the ? symbol will open a TCP connection to it. If the instance accepts incoming connections on that port, a success message is displayed as shown in Figure 25.

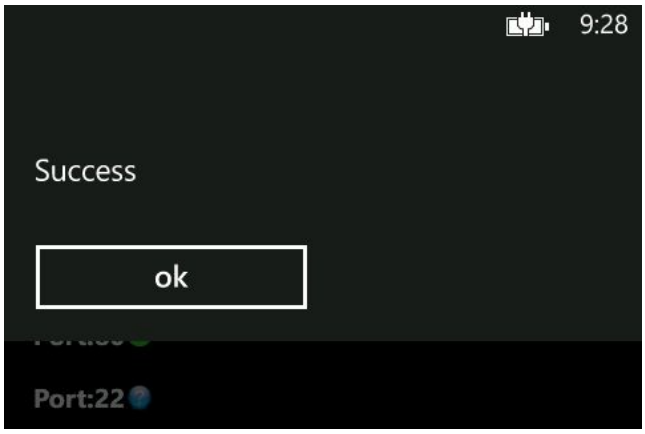

Figure 25: Success Message for Opening TCP Connection

If the port responded, the icon is changed to a tick mark as shown in Figure 26.

Figure 26: Icon showing confirmed and unconfirmed open port

Manually confirming unexpected open ports is a faster option for large AWS deployments rather can attempting open TCP connections with thousands of instances.

Report Summary in Live Tile

The Live Tile of the AWSSec app shows the number of accounts configured. The Live Tile will “flip” periodically to show the violation count from the previous scan.

Figure 27 illustrates the main live tile indicating that waccounts are configured in the app.

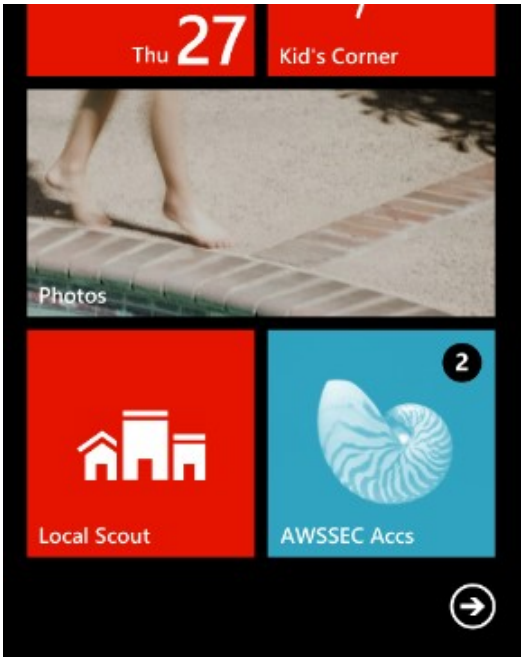

Figure 27: Live Tile showing the number of accounts configured

The flipped tile shows the number of issues identified by the last scan. Figure 28 shows that the app found no security warnings in the previous scan.

Figure 28: Live Tile showing violation count from the previous scan

The number of issue shown in the flip tile is the sum of all issues from all accounts scanned in the previous run.

6.5.4 Deployment

The team did not publish the tool in the Windows Phone Store. The projects that wanted to use the tool had to developer unlock the phone and install the package manually.

The tool does not have update mechanism to upgrade itself to newer versions. Presently, the upgrades must be manually checked by the project teams and performed. It is recommended that the future versions of the tool be either published to Windows Store (as a private app) or provide automatic update checking feature.

6.6 Automated Audits by Cloud Services Team

Cloud Services team also followed DevOps and automated many of its policy audits. Its automated audits logged policy violations and notified affected project teams periodically. Some of the audits that were done by the cloud services team were:

- Verifying that every IAM username in the organization's AWS infrastructure follows the cloud IAM naming standard. This naming standard differentiated between people and service user accounts.
- All IAM people user accounts have equivalent username in the corporate Active Directory. This ensured correlating IAM users with employees.
- Verify that all people user accounts have Multi Factor Authentication enabled.
- Verify that the IAM user account password are changed periodically.
- Report dormant IAM user accounts.

7 Conclusion

This research established that security automation and security self-certification were essential components of successful security governance in large scale cloud DevOps deployments.

As organization adopted agile methodology, the security tasks are also required to be agile. Given the shortage of information security experts in any given organization, it is also train the DevOps teams for security self-certification and provide them with automated security tools.

There are no recommendations on which security workflow tasks must be automated and which should be allowed for self-certification. In this research, the teams were provided with different self-certification tasks depending on the skills within the team and the business criticality of their product.

The findings of the present study would recommend to begin automation by identifying security tasks that are easily delegated to the teams themselves. DevOps relies on automation, and so work with the teams to automate core security tasks.

It is highly recommended to perform a gap assessment of all security tasks under existing security governance model. The objective of the gap assessment is to identify

- Security processes that are unsuitable for agile and cloud
- Automation opportunities
- Security tasks that can be delegated to the project teams
- Training needs for teams in preparation for security self-certification

Automation and self-certification do not imply that the security team has a fully hands-off approach to security governance. Security experts must be made available during the key phases of agile development to guide the teams.

8 Bibliography

[1] Amazon Web Services, “Auditing Security Checklist for Use of AWS,” June 2013. [Online]. Available: http://media.amazonwebservices.com/AWS_Auditing_Security_Checklist.pdf.

[2] Amazon Web Services, “Security Resources,” [Online]. Available: http://aws.amazon.com/security/security-resources/. [Accessed 2014].

[3] Amazon Web Services, “Security at Scale: Governance in AWS,” October 2015. [Online]. Available: https://d0.awsstatic.com/whitepapers/compliance/AWS_Security_at_Scale_Governance_in_AWS_Whitepaper.pdf.

[4] Cloud Security Alliance, “Security Guidance for Critical Areas of Focus in Cloud Computing,” 11 November 2011. [Online]. Available: https://downloads.cloudsecurityalliance.org/initiatives/guidance/csaguide.v3.0.pdf.

[5] ISACA, “Information Security Governance for Board of Directors and Executive Management 2nd Edition,” 2006. [Online]. Available: http://www.isaca.org/knowledge-center/research/documents/information-security-govenance-for-board-of-directors-and-executive-management_res_eng_0510.pdf.

[6] NIST, “NIST SP 800-100, Information Security Handbook: A Guide for Managers,” Oct 2006. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-100.pdf.

[7] IBM, “What is cloud computing?,” IBM, [Online]. Available: https://www.ibm.com/cloud-computing/learn-more/what-is-cloud-computing/. [Accessed 12 March 2017].

[8] NIST, “The NIST Definition of Cloud Computing,” [Online]. Available: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf.

[9] E. C.-L. Raghu Yelluir, Building the Infrastructure for Cloud Security, Apress, 2014.

[10] S. W. James Shore, The Art of Agile Development, O'Reilly Media, 2007.

[11] Amazon Web Services, “AWS Security Best Practices,” August 2016. [Online]. Available: http://aws.amazon.com/jp/whitepapers/aws-security-best-practices/.

[12] P. D. Steve Morad, “Operational Checklists for AWS,” June 2013. [Online]. Available: http://media.amazonwebservices.com/AWS_Operational_Checklists.pdf.

[13] T. Stickle, “Secure Microsoft Application on AWS,” 1 Aug 2012. [Online]. Available: http://aws.amazon.com/whitepapers/secure-microsoft-applications-on-aws/.

[14] J. Varia, “Architecting for the Cloud: Best Practices,” January 2011. [Online]. Available: http://media.amazonwebservices.com/AWS_Cloud_Best_Practices.pdf.

[15] Microsoft, “Managing Security and Privacy within DevOps,” June 2015. [Online]. Available: http://download.microsoft.com/download/9/2/4/924BCEEE-A3BA-4DB9-990F-F2A34DFC7E72/3851_Managing-Security-and-Privacy-within-DevOps_Article.docx.

[16] A. N. Carlos Conde, “Development and Test on Amazon Web Services,” November 2012. [Online]. Available: http://media.amazonwebservices.com/AWS_Development_Test_Environments.pdf.

[17] National Security Agency, “Guide to the Secure Configuration of Red Hat Enterprise Linux 5,” 20 December 2007. [Online]. Available: https://centoshelp.org/docs/RHEL-Guide-i731.pdf.

APPENDIX 1: Hardening Script for Cent OS and RHEL

The top few lines of the Perl hardening script developed by the author of this thesis is provided below.

```
#!/usr/bin/perl
use strict;
use Getopt::Long;

# ~+~+~+~+~+
# http://www.nsa.gov/ia/_files/os/redhat/rhel5-guide-i731.pdf
# ~+~+~+~+~+

# Set internal PATH to use sbin binaries
$ENV{'PATH'}=$ENV{'PATH'}.":/sbin:/usr/sbin";

my $debug = 1;
my $is_normal_user = `id -u`; chomp $is_normal_user;
my %report;
my $output_mode = "screen";

# print to act like println
$\ = "\n";

my %available_audits = (
    "partitions" => \&check_partitions,
    "network" => \&check_network,
    "pkgintegrity" => \&check_pkgintegrity,
    "aide" => \&check_aide,
    "fileownership" => \&check_fileownership,
    "exploitprotection" => \&check_exploitprotection,
    "useraccs" => \&check_useraccs,
    "sshd" => \&check_sshd,
    "runningservices" => \&check_runningservices,
    "all" => \&check_all);
```

-- END OF CODE SNIPPET --

APPENDIX 2: Security Patch Checking Script for Cent OS

The top few lines of the python script for security patch checking is provided below.

```
#!/usr/bin/python
from datetime import date, datetime, timedelta
from time import mktime
from optparse import OptionParser
from urllib2 import urlopen, HTTPError, URLError
import gzip, mailbox, email, os, sys, re, commands, time, errno, plat-
form

# Allowed CentOS releases and their release dates
# Release dates are used to speed up downloads
# by starting only from the release date.
# Get the release dates from:
# http://en.wikipedia.org/wiki/CentOS#Release_history
centos_release_dates = {}
centos_release_dates[6] = [2011, 7]
centos_release_dates[5] = [2007, 4]
centos_mailinglist_url = "http://lists.centos.org/pipermail/centos-an-
nounce/"

# How many days old AMI can be used?
CLOUDSERVICES_AMI_MAX_LIFETIME = 180
CLOUDSERVICES_AMI_RELEASE = "/etc/cloudservices-ami-release"

REGEX_SLASH = re.compile(r"/")
REGEX_SPACE = re.compile(r" ")
REGEX_SPACES = re.compile(r'\s+')
REGEX_RISK = re.compile(r".*(Critical|Important|Moderate|Low).*")
REGEX_RISK_INFO = re.compile(r".*(https?:\/\/rhn.redhat.com/er-
rata/RH[SB]A.+\.html).*")

# http://www.rpm.org/max-rpm/ch-rpm-file-format.html
# The format is: pkgname-version-release.arch.rpm
REGEX_RPMFILENAME = re.compile(
    r"(.*)-([\d\.]+)-(.*)\.(i[3456]86|x86_64|ia64|s390|s390x|alpha|no-
arch|src)\.rpm")


def download_updates(os_ver):
    security_patches = set()
    _begin_year = centos_release_dates[os_ver][0]
    _begin_mon = centos_release_dates[os_ver][1]
```

```
try:
    FILE_OUT = open(_out_filename, "w")
except IOError, (ex_no, ex_str):
    fatal_error(
```

-- END OF CODE SNIPPET –

APPENDIX 3: List of Figures

Figure 1: Security Governance Process 2
Figure 2. AWS Regions and Availability Zones 10
Figure 3: Sample Web Service Architecture in AWS 12
Figure 4. DevOps 15
Figure 5: Cloud Adoption Strategy 18
Figure 6: Secure Scrum 19
Figure 7. Security tasks for every sprint 20
Figure 8: Cloud Account Management within the Organization 22
Figure 9: Data Classification 23
Figure 10: AWS Encryption Policy 24
Figure 11: Network Scanning Deployment 28
Figure 12: Python AWSSec output showing a violation 30
Figure 13: AWSSec Tool Tasks 31
Figure 14: Multiple Accounts in AWSSec 32
Figure 15: Add Account Screen of AWSSec Tool 33
Figure 16: Scan Settings - Region Selection 34
Figure 17: Scan Settings Screen 35
Figure 18: AWSSec Scan in Progress. 36
Figure 19: Account with no violations. 37
Figure 20: List of Scan Violations 38
Figure 21: Scan report showing multiple violations (in prototype UI) 39
Figure 22: Instance Details in Violation Report 40
Figure 23: Not Applying Corp IP Security Group is a violation 41
Figure 24: Ports tab showing two possibly open ports 42
Figure 25: Success Message for Opening TCP Connection 42
Figure 26: Icon showing confirmed and unconfirmed open port 43
Figure 27: Live Tile showing the number of accounts configured 44
Figure 28: Live Tile showing violation count from the previous scan 44