|  |  |
| --- | --- |
| Pilot AWS Automation – EC2DataTransfer  Technote  Date: 17/10/2024 Author: Neal Ward  0.1  Internal |  |

Contents

[1 Background 3](#_Toc180090605)

[1.1 Scope 3](#_Toc180090606)

[1.2 Overview 3](#_Toc180090607)

[2 Design 3](#_Toc180090608)

[2.1 Infrastructure 3](#_Toc180090609)

[2.2 Workflow 5](#_Toc180090610)

[3 Implementation 6](#_Toc180090611)

[Appendix A – Sandbox Configuration 8](#_Toc180090612)

[SGN-OPS-EC2DataTransfer-AutomationRole 8](#_Toc180090613)

[SGN-GuardDuty-EC2Staging-ProtectionRole 8](#_Toc180090614)

[sgn-ops-ec2staging S3 Bucket policy 10](#_Toc180090615)

[EC2 Profile Role 10](#_Toc180090616)

Table of Figures

[Figure 1 - Proposed Design 4](#_Toc180090617)

[Figure 2 - Graph workflow view 5](#_Toc180090618)

**Version Control**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description |
| 0.1 | 17/10/2024 | Neal Ward | Initial draft for review |

**Reviewers**

|  |  |
| --- | --- |
| **Name** | **Role** |
| Alex Delmaine | Security Architect |
| Gary Piper | AWS Consultant |
| Graham Beer | AWS Consultant |
| Kevin Brady | SGN Project Manager |
| Olu Adejoro | SGN Security Analyst/ Consultant |
| Lloyd Newton | SGN Security Analyst/ Consultant |
| Matt Barrow | CGI AWS SME |
| Santosh Naik | IAM Security SME |

**Approvers**

|  |  |
| --- | --- |
| **Name** | **Role** |
| Stephen Morrison | InfoSec Manager |
| Jake Kerr | Cloud Architect |
| Anthony Hobson | IT Technical Design Architect |
| Alan Gordon | AWS Platform Manager |

# Background

SGN does not have a sanctioned secure channel for transferring files to and from EC2s. As a result, various means are used, some of which are insecure and highly risky, and expose SGN to malware propagation, data exfiltration and server compromise. This proposal is to implement a secure mechanism for transfer of files using simple AWS serverless resources and IAM access controls.

The CR2261 AWS Security Optimisation project is implementing this solution as part of its automation and segmentation initiatives.

These reasons file transfers are being used at all (rather than deployment pipelines) may not be justified, which is a wider issue to resolved, however this secure transfer channel will establish audit, security and control quickly while the wider issue is resolved and thus mitigate the current risk. This may ultimately make this a tactical solution with a strategic position of a break-glass mechanism only. This is the be decided as part of the subsequent design documentation

## Scope

* Corporate AWS Environments
* AWS Accounts (for pilot)
  + AWS SGN DevTest T1 Ireland
  + AWS SGN DevTest T2 Ireland
* Replacement of insecure channels used today to transfer data to and from EC2s

## Overview

The proposal is to migrate the prototype secure transfer mechanism that has been tested in the AWS Sandbox to DevTest T1 and T2 to pilot in a more formal and utilised environment. The pilot would achieve the following goals:

* Formally test the prototype in a more formal and diverse environment
* Refine the workflow based on stakeholder feedback
* Integrate with SGNs Teams for notification of approvals
* Evaluate if centralised deployment and execution is feasible
* Establish the target operating model (approver lists and thresholds for high risk approvals)

Upon successful completion of the pilot, the following will be produced:

* Decision as to how this mechanism would be employed going forward
* Design document for the SGN-OPS-EC2DataTransfer Automation for TDA approval
* Technote describing the insecure channels for transfer that should be decommissioned if this mechanism is adopted strategically.

The pilot is expected to run for 2 months to provide adequate time for it to be used by SGNs support partner, refine where required and produce the documents.

# Design

The design is summarised in this technote and will be described in detail in the subsequent design documents put forward to the TDA upon successful completion of the pilot.

## Infrastructure

AWS services are used for all aspects of the design and all aspects of this design are serverless.

### Current Architecture

Currently a variety of methods are employed to move data to and from EC2s, all of which are insecure:

* BAUADMIN server, acting as a bastion, with the DFS legacy file share SMB connected to get the files onto that server, then using SSH and RDP from there to send/ receive the files to the target EC2
* RDP and SSH used from workspaces to directly send/ receive files to EC2 servers
* Ad-hoc AWS cp API from variety of S3 buckets to EC2s, where EC2 has S3 access

SGNs support supplier is using these insecure channels for:

* Take application log files from application servers.
* Install cert files onto application servers.
* Install new software onto application servers.

### Proposed Design

The following diagram shows how Systems Manager Automation can provide a controlled and secure transfer of data to and from EC2s.

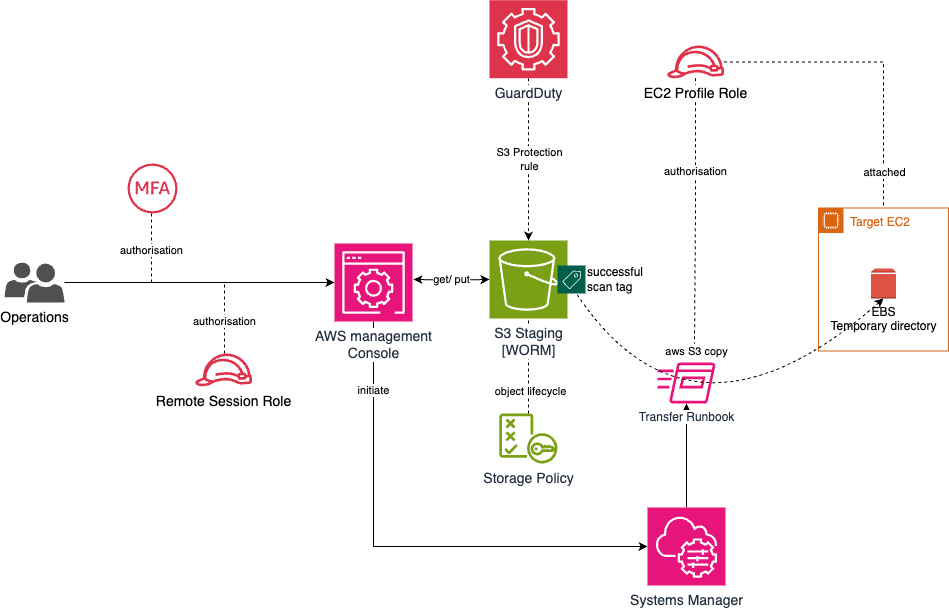


Figure - Proposed Design

#### Main Components

The following AWS components form the core of the design. Subsequent documents will provide a more detailed design.

|  |  |
| --- | --- |
| **Systems Manager Automation** | **SGN-OPS-EC2DataTransfer** Systems Manager Automation Document is the main component that manages approval and workflow to add the Security group rule. |
| **Automation IAM Role** | **SGN-OPS-EC2DataTransfer-AutomationRole** AWS IAM role that provides the Automation Document permission to run the copy scripts on the target EC2 and write the reports to the S3 staging prefix  (See Appendix A for policy reference) |
| **EC2 Profile Policy** | **sgn-ops-ec2staging-policy**  Provides the EC2 with the permissions to read from the S3 staging bucket Inbound prefix and write to the outbound prefix. This needs to be attached to all EC2 profile Roles  (See Appendix A for policy reference) |
| **GuardDuty Scanning role** | **SGN-GuardDuty-EC2Staging-ProtectionRole**  This provides the permissions for GuardDuty to scan and tag the S3 staging bucket  (See Appendix A for policy reference) |
| **GuardDuty S3 Protection rule** | This directs GuardDuty to scan al objects that are written to the staging S3 bucket  (See Appendix A for policy reference) |
| **S3 Bucket** | **sgn-ops-ec2staging**  SNS topic to receive events (approvals and completion/ failure) of this automation, so that it can be integrated with the AWS Platform. It is proposed to make this bucket Write Once read Many (WORM) for non-repudiation of contents and to allow SecOps forensic analysis of the files if required. A bucket lifecycle will tier down the objects to cheaper storage.  (See Appendix A for bucket policy reference) |

#### Additional changes required

* Allow RemoteAccess permission set to discover and execute the automation document but not view the content.
* Allow AWS Platform team to modify the automation document.
* Allow SecOps Admin permission set to read malware tagged S3 Objects for copy to an analysis environment

## Workflow

This Systems Manager Automation Document facilitates file transfers between EC2 instances and a staging S3 bucket which will enforce non-repudiation and malware scanning. The automation supports both inbound (S3 to EC2) and outbound (EC2 to S3) transfers of multiple files using wildcards. It can be used for multiple EC2 instances running on Linux or Windows platforms.

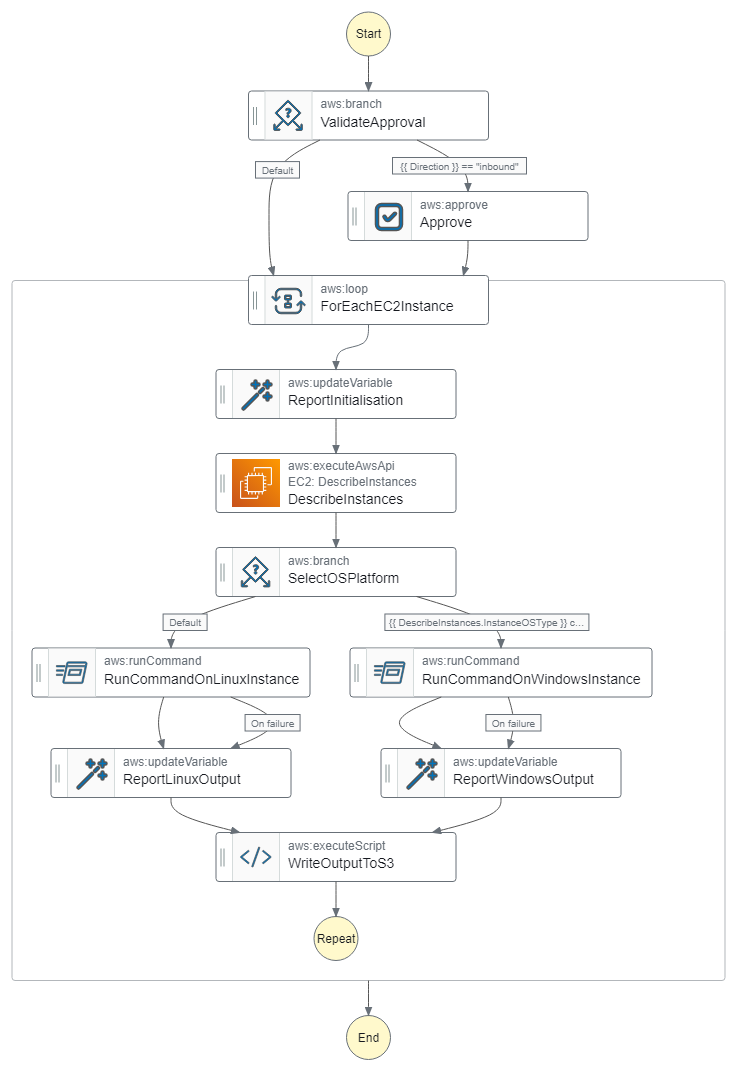


Figure - Graph workflow view

|  |  |
| --- | --- |
| **ValidateApproval** | Branch to Approve step if an inbound request to EC2. |
| **Approve** | Wait for Approval from a person in the specified approval roles before proceeding. |
| **ForEachEC2Instance** | Loops over each EC2 instance selected on execution and executes the transfer process against each. |
| **ReportInitialisation** | Initialises a report that will represent the detailed file transfer output for each EC2 instance. |
| **DescribeInstances** | Retrieves the OS type for each instance so the correct script can be executed on that instance. |
| **SelectOSPlatform** | Branch to the correct OS type to execute the correct format of script. |
| **RunCommandOn**  **LinuxInstance** | Executes the copy command using the AWS CLI:   * For inbound direction: Copies files from the S3 bucket to the EC2 instance’s local directory. * For outbound direction: Copies files from the EC2 instance to the S3 bucket.   Handles success and failure conditions, ensuring that the process exits cleanly on failure. |
| **RunCommandOn**  **WindowsInstance** | Runs the same logic as the Linux step but on windows EC2 servers |
| **WriteOutputToS3** | Writes the output report to the staging S3 bucket under the Reports folder. Each EC2 instance generates its own separate report, including execution ID and instance details. |

### Input Parameters

|  |  |
| --- | --- |
| **Direction** | Defines whether the copy is from S3 to EC2 (inbound) or from EC2 to S3 (outbound). |
| **FileName** | Specifies the file(s) to copy, supporting wildcards. |
| **SourcePath and TargetPath** | Defines the source and destination directories for the files. |
| **InstanceIds** | Specifies the EC2 instances for the copy operation. |
| **ChangeRequest** | CR that the transfer is required for |
| **Comment** | Reason for transfer |

# Implementation

The following activities would be carried out on DevTest T1 and T2 pilot implementation:

1. Create Systems Manager Document in both DevTest T1 and T2 accounts (copied from Sandbox)
2. Create IAM Automation role
3. Create the GuardDuty malware scanning role
4. Add the S3 access policy to the standard EC2 Profile roles or policy
5. Create SNS topic
6. Create/ modify SNS to Teams notification channel (exists for Security group change notification)
7. Create AWS Platform team role (for approvals and permission to bypass this automation)
8. Configure the rules in the Automation to prevent rule additions and flag high risk
9. Provide permissions in RemoteAccess role for SGNs support teams to list and execute the Systems Manager Document (specific to resource or could be conditional on Document tagging)

Once implemented, SGN can work with Enzen to make sure the workflow and process is not impeding their BAU support and refine as necessary over the 2-month pilot phase.

# Appendix A – Sandbox Configuration

These are the configurations that are being testing in the Sandbox and form the basis of the pilot resources.

## SGN-OPS-EC2DataTransfer-AutomationRole

### SGN-OPS-EC2DataTransfer-InlinePolicy

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "EC2RequiredPermissions",

"Effect": "Allow",

"Action": "ec2:DescribeInstances",

"Resource": "\*"

},

{

"Sid": "SystemsManagerRequiredPermissionsToRunCommandsOnEC2",

"Effect": "Allow",

"Action": [

"ssm:DescribeInstanceInformation",

"ssm:SendCommand",

"ssm:ListCommands",

"ssm:GetCommandInvocation",

"ssm:ListCommandInvocations"

],

"Resource": "\*"

},

{

"Sid": "S3RequiredPermissionsToReportOutcome",

"Effect": "Allow",

"Action": [

"s3:PutObject",

"s3:PutObjectAcl"

],

"Resource": "arn:aws:s3:::sgn-ops-ec2staging/Reports/\*"

}

]

}

## SGN-GuardDuty-EC2Staging-ProtectionRole

### SGN-GuardDuty-EC2Staging-InlinePolicy

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "AllowManagedRuleToSendS3EventsToGuardDuty",

"Effect": "Allow",

"Action": [

"events:PutRule",

"events:DeleteRule",

"events:PutTargets",

"events:RemoveTargets"

],

"Resource": [

"arn:aws:events:eu-west-1:103429827345:rule/DO-NOT-DELETE-AmazonGuardDutyMalwareProtectionS3\*"

],

"Condition": {

"StringLike": {

"events:ManagedBy": "malware-protection-plan.guardduty.amazonaws.com"

}

}

},

{

"Sid": "AllowGuardDutyToMonitorEventBridgeManagedRule",

"Effect": "Allow",

"Action": [

"events:DescribeRule",

"events:ListTargetsByRule"

],

"Resource": [

"arn:aws:events:eu-west-1:103429827345:rule/DO-NOT-DELETE-AmazonGuardDutyMalwareProtectionS3\*"

]

},

{

"Sid": "AllowPostScanTag",

"Effect": "Allow",

"Action": [

"s3:PutObjectTagging",

"s3:GetObjectTagging",

"s3:PutObjectVersionTagging",

"s3:GetObjectVersionTagging"

],

"Resource": [

"arn:aws:s3:::sgn-ops-ec2staging/\*"

]

},

{

"Sid": "AllowEnableS3EventBridgeEvents",

"Effect": "Allow",

"Action": [

"s3:PutBucketNotification",

"s3:GetBucketNotification"

],

"Resource": [

"arn:aws:s3:::sgn-ops-ec2staging"

]

},

{

"Sid": "AllowPutValidationObject",

"Effect": "Allow",

"Action": [

"s3:PutObject"

],

"Resource": [

"arn:aws:s3:::sgn-ops-ec2staging/malware-protection-resource-validation-object"

]

},

{

"Effect": "Allow",

"Action": [

"s3:ListBucket"

],

"Resource": [

"arn:aws:s3:::sgn-ops-ec2staging"

]

},

{

"Sid": "AllowMalwareScan",

"Effect": "Allow",

"Action": [

"s3:GetObject",

"s3:GetObjectVersion"

],

"Resource": [

"arn:aws:s3:::sgn-ops-ec2staging/\*"

]

},

{

"Sid": "AllowDecryptForMalwareScan",

"Effect": "Allow",

"Action": [

"kms:GenerateDataKey",

"kms:Decrypt"

],

"Resource": "arn:aws:kms:eu-west-1:103429827345:key/3f12d014-4db6-4f33-89a8-4e1695dc9ab5",

"Condition": {

"StringLike": {

"kms:ViaService": "s3.\*.amazonaws.com"

}

}

}

]

}

## sgn-ops-ec2staging S3 Bucket policy

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "NoReadExceptForClean",

"Effect": "Deny",

"Principal": "\*",

"Action": [

"s3:GetObject",

"s3:GetObjectVersion"

],

"Resource": [

"arn:aws:s3:::sgn-ops-ec2staging",

"arn:aws:s3:::sgn-ops-ec2staging/\*"

],

"Condition": {

"StringNotLike": {

"aws:PrincipalArn": "arn:aws:iam::103429827345:role/SGN-GuardDuty-EC2Staging-ProtectionRole"

},

"StringNotEquals": {

"s3:ExistingObjectTag/GuardDutyMalwareScanStatus": "NO\_THREATS\_FOUND"

}

}

},

{

"Sid": "OnlyGuardDutyCanTag",

"Effect": "Deny",

"Principal": "\*",

"Action": "s3:PutObjectTagging",

"Resource": [

"arn:aws:s3:::sgn-ops-ec2staging",

"arn:aws:s3:::sgn-ops-ec2staging/\*"

],

"Condition": {

"StringNotLike": {

"aws:PrincipalArn": "arn:aws:iam::103429827345:role/SGN-GuardDuty-EC2Staging-ProtectionRole"

}

}

}

]

}

## EC2 Profile Role

### sgn-ops-ec2staging-policy

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "InboundStagingAccess",

"Effect": "Allow",

"Action": [

"s3:GetObject",

"s3:GetObjectAcl"

],

"Resource": [

"arn:aws:s3:::sgn-ops-ec2staging/Inbound/",

"arn:aws:s3:::sgn-ops-ec2staging/Inbound/\*"

]

},

{

"Sid": "BucketStagingAccess",

"Effect": "Allow",

"Action": [

"s3:ListBucket"

],

"Resource": [

"arn:aws:s3:::sgn-ops-ec2staging"

]

},

{

"Sid": "OutboundStagingAccess",

"Effect": "Allow",

"Action": [

"s3:PutObject",

"s3:PutObjectAcl"

],

"Resource": [

"arn:aws:s3:::sgn-ops-ec2staging/Reports/",

"arn:aws:s3:::sgn-ops-ec2staging/Reports/\*"

]

}

]

}