

# Virtual Research Environment INSTALLATION GUIDE

---

[hello@foodagility.com](mailto:hello@foodagility.com)



This code is provided under a Creative Commons Share Alike licence

#### Disclaimer

*The sample code and scripts to create the example virtual research environment is provided “as is” and any express or implied warranties, including the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall Food Agility CRC or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) sustained by you or a third party, however caused and on any theory of liability, whether in contract, strict liability, or tort arising in any way out of the use of this sample code, even if advised of the possibility of such damage.*

*As with all technology products and environments, advances in technology and threats to online technology are ever changing, as such there is no assurance or guarantee that the use of the example virtual research environment will provide the user with their required security for real life online environments. The use of and any application of this example in whole or in part to real life environments, whether now or in the future, is done so at the sole risk of the user.*

# Table of Contents

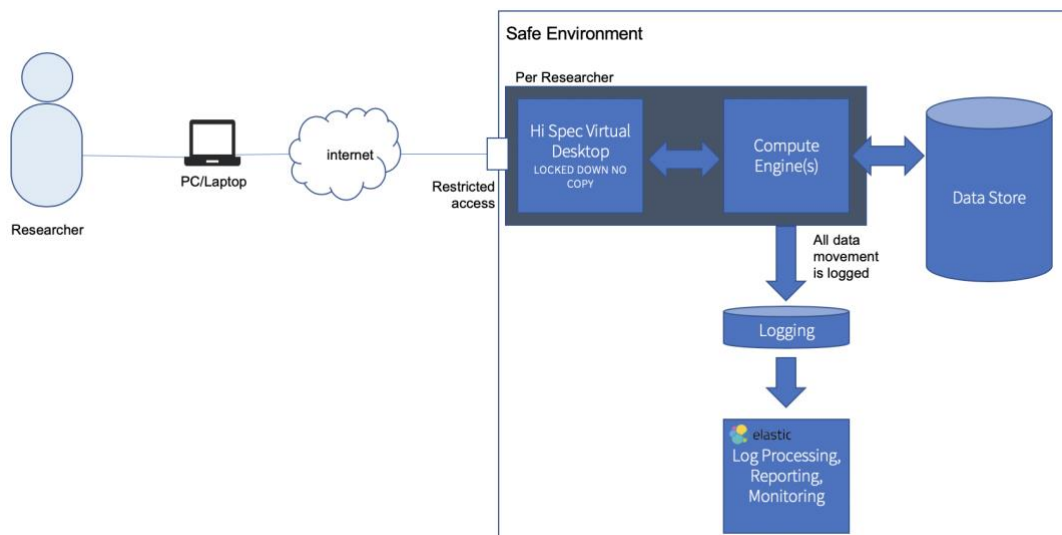
<b>1. INTRODUCTION</b>	<b>4</b>
<b>2. ASSUMED KNOWLEDGE</b>	<b>6</b>
<b>3. ARCHITECTURE</b>	<b>7</b>
3.1 OVERVIEW OF AWS WORKSPACES ARCHITECTURE	7
3.1.1 NETWORK AND SYSTEMS ARCHITECTURE OF AWS WORKSPACES	8
3.2 OVERVIEW OF ELK STACK	9
3.2.1 NETWORK AND SYSTEMS ARCHITECTURE OF ELK STACK	9
4.1 INTRODUCTION	10
4.2 PREPARATION	10
4.3 CREATING THE ENVIRONMENT	10
4.3.1 CORE NETWORK INFRASTRUCTURE	10
4.3.2 ADMIN VM	11
4.3.3 AWS WORKSPACES	12
4.3.4 DATASCIENCE COMPUTE	14
4.3.5 DATASCIENCE STORAGE	15
4.3.6 ELK Stack – (ElasticSearch LogStash Kibana)	15
<b>5. ADMINISTRATION</b>	<b>18</b>
5.1 ADDING USERS	18
5.2 ADDING RESEARCH VXDS	20
5.3 ADDING DATA SETS	20
5.4 ELK EXAMPLES - TRACKING UNDESIRABLE ACTIVITY	20
<b>6. EXTENSIONS</b>	<b>23</b>
6.1 ALLOWING ACCESS TO MULTIPLE VMS FROM THE VXD	23
6.2 MFA INTEGRATION WITH AWS WORKSPACES AND OPENVPN	23
6.3 CONFIGURATION OF REMOTE SUPPORT SERVICE FOR AWS WORKSPACE	23
6.4 OTHER	23

# 1. INTRODUCTION

---

This document provides an architectural overview and installation guide for the Food Agility's Virtual Research Environment.

This sample system provides an environment that enables safe remote access to sensitive data. It provides an example of how to enable researchers or any external agent to remotely access compute and data resources whilst ensuring that it is difficult to copy sensitive data out of the environment. It's important to acknowledge that it is impossible to ever completely prevent data theft where someone has access to that data. The default option in this solution is acceptable in a reasonably trusted environment but can be beefed up considerably if required.



Creation of the system is scripted. The environment runs in the AWS cloud. It provides access via VPN to a remote desktop system which has access to a simple compute environment which leverages data from an AWS S3 bucket. All systems log activity to an Elastic instance (Elastic Search, Logstash and Kibana) to enable audit and alerting to ensure any nefarious activity can be detected. The remote desktop has been designed to be restricted from allowing data to be copied out.

This example intentionally uses comparatively simple technologies. This is important as most of the components are well understood and most DevOps will be comfortable working with this. Also, most businesses will be familiar with these concepts and tools and so it will be easier for businesses to adopt it.

This document provides:

- Architectural Overview;
- Usage Guide; and
- Examples of possible extensions.

It is important to recognise that this example is just that – *an example only*. It is not earth-shattering technology. It should be seen as a “leg up” for anyone wanting to achieve the same goal. Anyone using this example code needs to carefully consider and review their implementation. We recommend penetration testing any solution based on this example code.

The benefits of this solution are:

- Valuable data held in a controlled safe environment
- Researchers access the system via web based remote desktop and VPN
- Remote desktop can be standardised and locked down to suit needs of the organisation and the risk levels of the data
- Compute resources are cloud based and offer a large variety of tools that can be run scalably to manage costs
- Access is controlled using Active Directory services and can be extended to use 2 factor authentication for added security
- Logging, Audit and Alerting capability for detection of data copying (and other activity as needed)
- Out of the box sample in AWS, scripted creation of the solution

## 2. ASSUMED KNOWLEDGE

---

The deployment of this solution requires knowledge the following:

1. AWS VPC, Workspaces, EC2, S3, Directory Service, ECS, ALB
2. Active Directory
3. DNS
4. ElasticSearch
5. Kibana
6. Docker
7. OpenVPN

Successful deployment of this architecture also requires knowledge of networking, specifically, existing networking architecture of the environment this service is being deployed to.

## 3. ARCHITECTURE

---

### 3.1 OVERVIEW OF AWS WORKSPACES ARCHITECTURE

This architecture overview references the AWS Workspaces architecture in 3.1.1

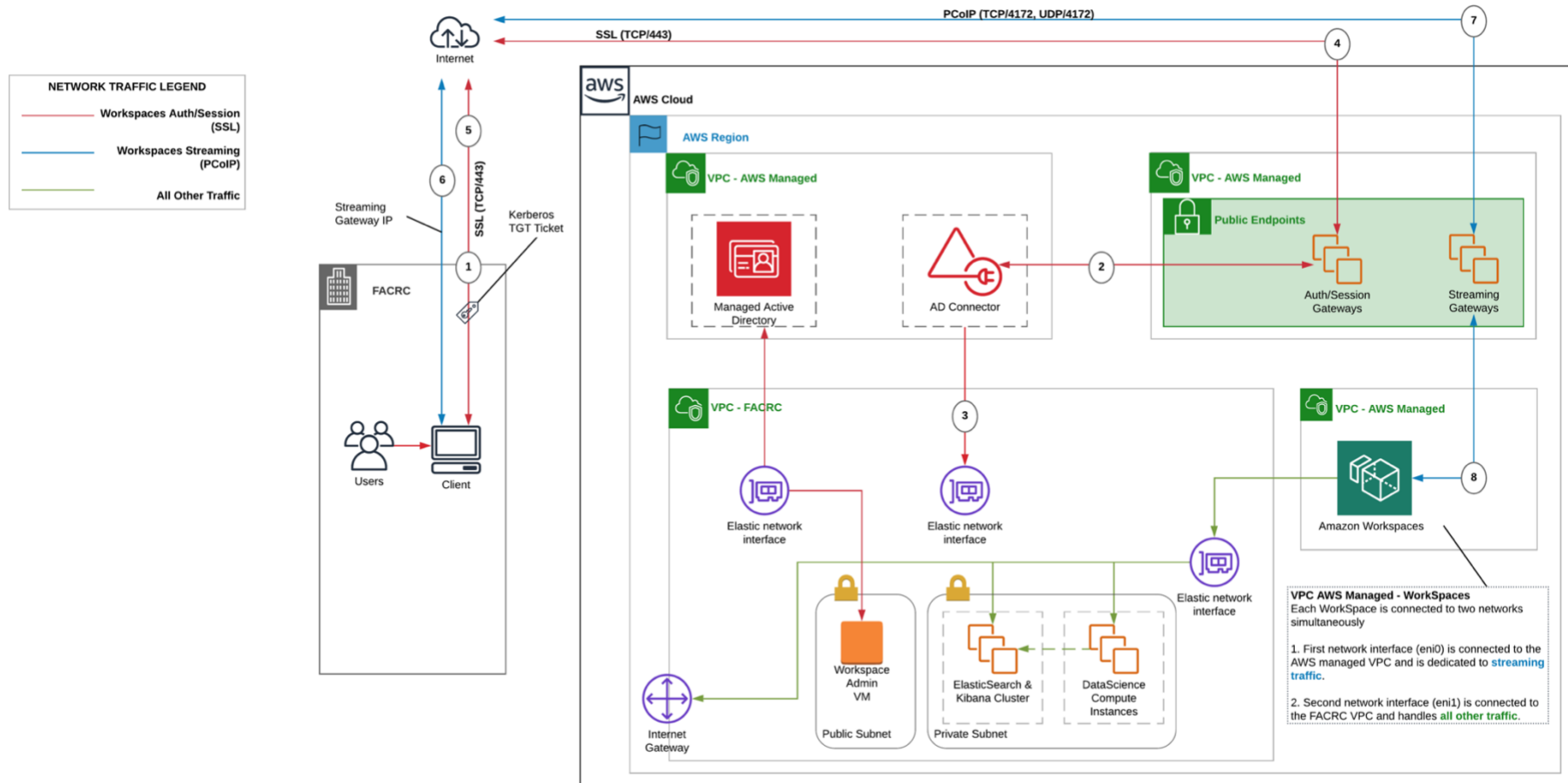
This architecture has AD DS deployed in the AWS Cloud in a standalone isolated environment. AWS Directory Service is used exclusively in this design. Instead of fully managing AD DS, AWS Directory Service performs tasks such as building a highly available directory topology, monitoring domain controllers, and configuring backups and snapshots. All of which would need to be self-managed if a traditional AD DS was deployed.

The AD DS (Microsoft AD) is deployed into dedicated subnets that span two Availability Zones, making AD DS highly available in the AWS Cloud. In addition to Microsoft AD, AD Connector is deployed for WorkSpaces authentication, this can also be used for MFA (not covered in this deployment) This ensures separation of roles or functions within the Amazon VPC, which is a standard best practice. This architecture uses the following components or constructs:

- Amazon VPC: Creation of an Amazon VPC with at least four private subnets across two Availability Zones (two for AD DS Microsoft AD, two for AD Connector or WorkSpaces).
- DHCP options set: Creation of an Amazon VPC DHCP options set. This allows defining a specific domain name and DNS(s) (Microsoft AD)
- AWS Directory Service: Microsoft AD deployed into a dedicated pair of VPC subnets (AD DS Managed Service).
- AWS Directory Services: AD Connector is deployed into a pair of Amazon VPC private subnets.
- Amazon WorkSpaces: WorkSpaces are deployed into the same private subnets as the AD Connector.

In summary this architecture doesn't have issues with reliance on connectivity to on-premises data center, latency, or data egress transfer costs (except where internet access is enabled for WorkSpaces within the VPC) because, by design, this is an isolated or cloud-only design.

### 3.1.1 NETWORK AND SYSTEMS ARCHIECTURE OF AWS WORKSPACES



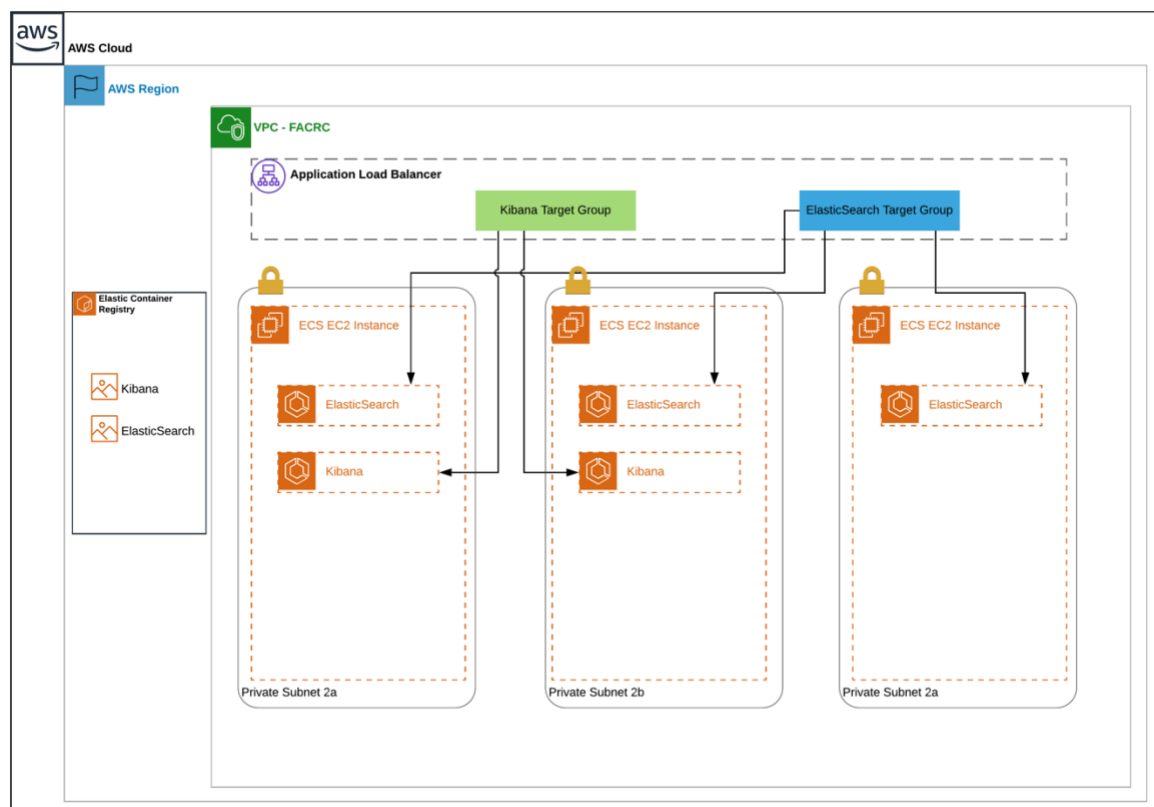


## 3.2 OVERVIEW OF ELK STACK

This architecture overview references the ELK stack architecture in 3.2.1

The design of the ELK stack revolves around a key concept of immutable infrastructure. All configuration is maintained in IaC (infrastructure as Code) and configuration changes result in the stack being redeployed. Data is persisted in EBS (Elastic Block Storage) volumes within the ECS (Elastic Container Service) stack. To provide high availability and load balancing across multiple Availability Zones, an Application Load Balancer is utilised to direct requests between available ECS instances for the respective services. Three ElasticSearch containers are deployed to ensure high availability. It is important to note that an odd number of instances is recommended to avoid split-brain situations in the cluster. Kibana is deployed in 2 pods for high availability reasons as well.

### 3.2.1 NETWORK AND SYSTEMS ARCHIECTURE OF ELK STACK



## 4. USER GUIDE

---

### 4.1 INTRODUCTION

This guide will outline the steps required to deploy the AWS secure working environment. The instructions will cover deployment of the following components:

1. Core Network Infrastructure
2. AWS Simple Directory Service
3. EC2 Instance for managing AWS Simple Directory Service and DNS
4. Deployment of an AWS Workspace Instance
5. Deployment of a Datascience Compute VM
6. Deployment of ElasticSearch and Kibana for logging and monitoring

### 4.2 PREPARATION

Prior to deployment of the stack the following pre-requisites are required:

1. AWS CLI - <https://aws.amazon.com/cli/>
2. Windows Subsystem for Linux (if running scripts in Windows environment) - <https://docs.microsoft.com/en-us/windows/wsl/install-win10>

### 4.3 CREATING THE ENVIRONMENT

#### 4.3.1 CORE NETWORK INFRASTRUCTURE

**Path:** workspaces/1-AWSWorkspacePreReq

The Core Infrastructure deploys the following services:

1. AWS Managed Directory Name
2. VPC Network and 4 Subnets (2 x Private & 2 x Public Subnets)

If the CIDR range needs to be modified edit `awsworkspace-prereq-parameters.json` and edit the following parameter values:

1. **VPCCIDRBlock** – A /16 CIDR address range you'd want your VPC to have
2. **PubSubnetA** – A CIDR range under the /16 VPC to use for your Public Subnet in Availability Zone A
3. **PubSubnetB** - A CIDR range under the /16 VPC to use for your Public Subnet in Availability Zone B. Note this cannot overlap with the range defined in PubSubnetA

4. **PriSubnetA** - A CIDR range under the /16 VPC to use for your Private Subnet in Availability Zone A. Note this cannot overlap with the range defined in PubSubnetA/B
5. **PriSubnetB** - A CIDR range under the /16 VPC to use for your Private Subnet in Availability Zone B. Note this cannot overlap with the range defined in PubSubnetA/B or PriSubnetA.

As part of the deployment the Active Directory domain name can be customised as well as the Administrator password. Both these values are set in the same `awsworkspace-prereq-parameters.json` using the following parameter values:

1. **ADName** – This is the Directory Domain name you want to use for the AWS Simple Directory Service that is being deployed.
2. **DirAdminPass** – This is the password you would like to use for the Administrator account that is created when the directory service is stood up.

Once the values are set run `workspace-prereq-deploy.sh` to deploy the network infrastructure stack.

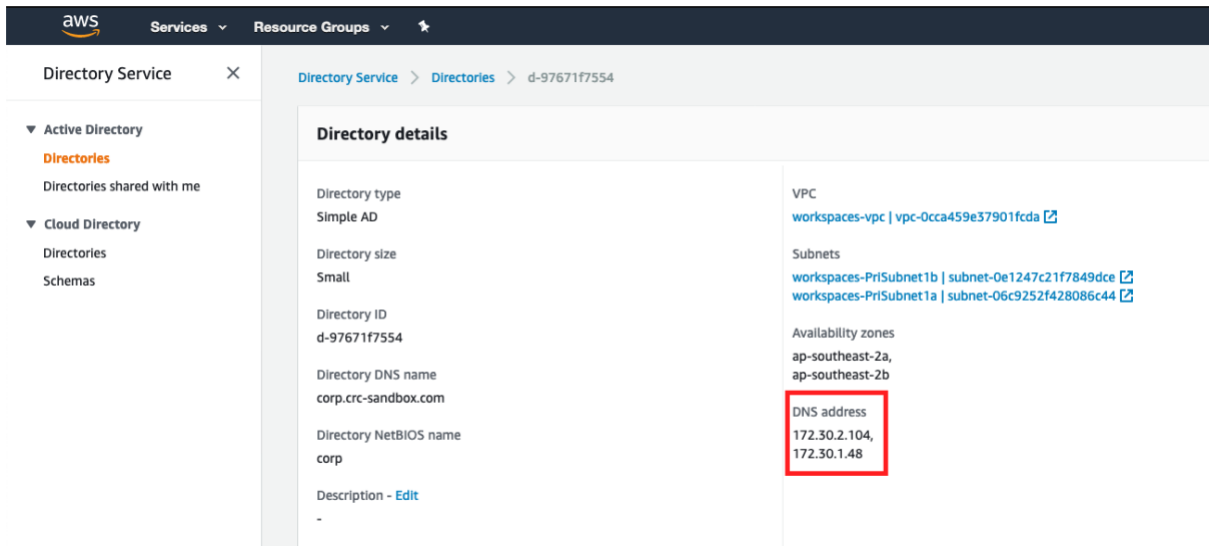
### 4.3.2 ADMIN VM

**Path:** `workspaces/2-AdminVM`

This deployment creates an EC2 instance in the public subnet of the VPC created in deployment #1. The VM runs Windows Server 2016 with Remote Server Administrative Tools (Active Directory Domain Services & DNS) installed. This VM can be used to manage users associated with the AWS Simple Directory service as well as create/update DNS entries associated to the domain. The VM is deployed with a Security Group entry allowing RDP from a whitelisted CIDR which is configurable in the Deployment parameters JSON file (`adminvm-params.json`). The following parameters are configurable:

1. **InstanceType**: This is the EC2 VM size you want to deploy.
2. **SourceCidrForRDP**: This is the Public IP CIDR you want to allow RDP access from
3. **PublicSubnet**: Choose one of the two public subnets IDs deployed in step #1
4. **VPC**: Enter the VPC ID that was deployed in step #1
5. **ADDirectoryId**: Enter the Directory ID of the AWS Simple Directory service that was deployed in Step #1
6. **ADDirectoryName**: Enter the Domain Name of the AWS Simple Directory that was deployed in Step #1

7. **ADDnsIpAddress1 & ADDnsIpAddress2:** This is the AWS provided DNS IPs after creation of the AWS Simple Directory in Step #1. To view this open the Directory Service section and view the details of the created directory. You'll see the DNS IPs outlined like the below example:



### 4.3.3 AWS WORKSPACES

**Path:** workspaces/3-AWWorkspacesDeploy

This deployment creates an AWS Workspace VM against the previously configured Directory and VPC for a given user. In a fresh environment there are no Workspace SOEs or bundles and therefore the initial deployment will create a base VM with Amazon's default Windows 10 SOE (excl. Office Suite). At time of writing the following Bundle ID `wsb-8vbljg4r6` will deploy a Windows 10 AWS Workspace in the `ap-southeast-2` region. Once the VM is deployed you are able to customise it as required and a new bundle can be created. When a new bundle is created the ID can be replaced in the `params.json` file for future deployments.

The following parameters can be edited as part of the `awsworkspace-params.json`:

1. **Bundle:** This is the bundle ID of the AWS Workspace VM which will be deployed into Workspaces
2. **Directory:** This is the ID of the AWS Simple Directory Service which was deployed as part of step 6.3.1
3. **User:** This is the username of the AWS Simple Directory user that the AWS Workspace will be deployed for.

## CUSTOMISING AWS WORKSPACE AND BUILDING A BUNDLE

Once the first AWS Workspace instance has been created use the AWS Workspace client (<https://clients.amazonworkspaces.com>) to connect to the instance.

## INSTALLING BASE SOFTWARE BUNDLE

Base software is installed using Chocolatey package management for Windows. A Powershell script that installs the service and associated software bundle is available in `workspaces/3-AWSWorkspaceDeploy/aws-workspace-chocolatey-install.ps1`. The software installed is:

1. R
2. R Studio
3. Pycharm Professional
4. Python 3.8.6

The list of software is defined in variable `$chocopackages` and can be modified to include other software available via the Chocolatey Package Repository <https://chocolatey.org/packages>. The command to execute this script and bypass execution policy is `powershell.exe -ExecutionPolicy Bypass -NoProfile -File .\aws-workspace-software.ps1`

## CONFIGURING WINLOGBEAT

A copy of the WinLog Beat YAML is provided in the following location of the deployment scripts `elk/3-ecs/winlogbeat.yaml`. Based on the provided YAML, edit lines 71 and 98 with the respective Kibana and Elasticsearch hostnames respectively. Once the lines are edited this file can be placed in the AWS Workspace under `C:\ProgramData\chocolatey\lib\winlogbeat\tools`

## CAPTURING AWS WORKSPACE IMAGE

Once all the required software is installed it is time to prepare the image for capture. To ensure the imaging process runs successfully Amazon has provided a Workspace Image Checker which tests the OS configuration is suitable for capture. This executable is located in `C:\Program Files\Amazon\ImageChecker.exe`. After the checker has run successfully perform the following steps on the AWS Console to capture the image:

1. Visit the AWS Workspaces section on the Console page and select the workspace you want to capture -> Actions -> Create Image
2. The following window will ask you to name the image and provide a description. It is a good idea to initiate a naming convention for such images so there are easy to identify and manage. **NOTE: The image capture process can take up to 45 minutes. During this time the AWS Workspace instance will be offline.**
3. Once the image is create it is time to create the Bundle. An AWS Workspace Bundle the collection of Image and compute tier that is deployed as an AWS Workspace

instance. To create a bundle select the Image that was created -> Actions -> Create Bundle

4. The next window will ask you the following parameters:
  - a. Bundle Name: This is the name of the bundle and what Administrators will see when creating an AWS Workspace instance via the Console
  - b. Description
  - c. Bundle Type: This is the CPU and Memory configuration of the Workspace instance. The CPU and Memory is predefined and non customisable. The following tiers are available:
    - i. Value: 1CPU 2GB Memory
    - ii. Standard: 2CPU 4GB Memory
    - iii. Performance: 2CPU 7.5GB Memory
    - iv. Power: 4CPU 16GB Memory
    - v. PowerPro: 8CPU 32GB Memory
    - vi. Root & User Volume: The following combinations are available to be used for your AWS Workspace:
    - vii. Root: 80GB User: 10GB
    - viii. Root: 80GB User: 50GB
    - ix. Root: 80GB User: 100GB
    - x. Root: 175GB User: 2000GB
    - xi. Root: 100GB User: 2000GB
5. Once the bundle has been create it can be used to deploy subsequent AWS Workspace VMs for users.

#### 4.3.4 DATASCIENCE COMPUTE

**Path:** datascience-compute/

The deployment of the data science compute instance involves the creation of a SSH Key Pair and deployment of an EC2 instance in a given VPC.

To deploy a data science EC2 instance run `create-sshkey.sh` first to deploy the EC2 key pair. Once this is deployed `dscicompute-deploy.sh` can be run. The following parameters can be customised as part of the deployment in the `datascience-compute-params.json`:

1. **SSHKeyName:** This is the name of the SSH Key Pair that was created as part of `create-sshkey.sh`
2. **ClientIPCIDR:** This CIDR range is added to the Security Group to allow where connections can be originated from within the AWS Workspaces VPC
3. **VPCID:** This is the VPCID in which you want the EC2 instance to be deployed into. This was created in Step 6.3.1
4. **AMI:** This is the Amazon Managed Image for the VM that is to be deployed. By default the following AMI will deploy a Ubuntu 18.04 Instance in the ap-southeast-2 region (`ami-08d1ad86b1c8269de`)

### 4.3.5 DATASCIENCE STORAGE

**NOTE: The S3 Bucket that is deployed doesn't allow public access to the contents stored. All requests must be authenticated.**

**Path:** datascience-storage/

For the purposes of this guide an S3 bucket is deployed for the purposes of DataScience storage. The S3 bucket is accessible using the IAM user belonging to the Account the Bucket is being deployed into.

To deploy the service simply run `s3bucket-deploy.sh`

The customisable parameter in this deployment is the **BucketPrefix** which is the prefixed name of the S3 bucket that is being deployed.

### 4.3.6 ELK Stack – (ElasticSearch LogStash Kibana)

**Path:** elk/

Deploying this stack comprises of three steps:

1. Deploying an Elastic Container Registry (ECR)
2. Building and pushing a docker image of ElasticSearch to ECR
3. Deploying the Elastic and Kibana stack

### DEPLOYING THE ELASTIC CONTAINER REGISTRY (ECR)

**Path:** elk/1-ecr

ECR is an AWS Service that stores container images in a centralised repository. For the purposes of this deployment ECR is being deployed to store the docker image for the ElasticSearch Service.

**Prerequisite:** An IAM user is required to be created in the AWS Account that is being deployed into. To create an IAM user please follow [https://docs.aws.amazon.com/IAM/latest/UserGuide/id\\_users\\_create.html#id\\_users\\_create\\_console](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users_create.html#id_users_create_console)

The following parameters can be customised in the `ecr-parameters.json` file

1. **AccountId:** This is the ID of the AWS account you are deploying the service into. To obtain the Account ID via the AWS CLI run `aws sts get-caller-identity`
2. **AWSECRAAdminUser:** This is the user that was created in the prerequisite step

## BUILDING ELASTICSEARCH DOCKER IMAGE

The dockerfile manifest is in `/elk/2-elasticsearch/Dockerfile` and contains the necessary parameters to build an ElasticSearch image. In the same folder is a shell script called `es-build.sh` which contains the steps run to build and publish the Docker image. At time of writing this guide the image built is ElasticSearch version 6.5.4. Parameters that can be customised as part of the build process are:

1. **AWS\_ACCOUNT\_ID**: This is the ID of the AWS account that contains the ECR created in step 6.3.5.1
2. **AWS\_DEFAULT\_REGION**: This is where you specify the region of where the ECR is deployed into. By default it is `ap-southeast-2`
3. **REPO\_NAME**: The name of the docker image you want to build, default is `elasticsearch`
4. **ES\_VERSION**: This is the version number of ElasticSearch you want to use as part of this Docker Image. Version 6.5.4 has been tested as stable.

To build the Docker image and publish it simply run `es-build.sh`

## DEPLOYING ELASTICSEARCH AND KIBANA

This step will deploy the ElasticSearch and Kibana services using Elastic Container Service (ECS). As part of the deployment the following parameters can be configured:

1. **AWSAccountId**: This is the ID of the AWS Account that the service is to be deployed into. Same as 6.3.5.2
2. **KeyName**: This is the AWS Key Pair to allow SSH connections to the underlying OS that is running in the ECS cluster.
3. **WorkspacesSG**: This is the Security Group used for AWS Workspaces.
4. **VPCId**: This is the VPC you want to deploy the stack into
5. **SubnetId**: The Subnet IDs that the ECS cluster will be deployed into
6. **DesiredCapacity**: This is the number of ECS compute nodes that will be deployed in the cluster. Three nodes is the recommended number for cluster stability.
7. **InstanceType**: This is the EC2 compute instance type each of the nodes will have. Note, if you choose an instance type too small, there will not be enough resources to run the ELK cluster reliably. Recommended InstanceType is `m5.4xlarge`
8. **ESInstanceCount**: The number of ElasticSearch containers to run in the cluster. This number should be the same number of nodes in the cluster (1 instance of ElasticSearch per node) to ensure availability.
9. **KibanaInstanceCount**: The number of instances of Kibana to run, default is 1

The output of the deployment will return 2 DNS values:

1. **ESALB (ElasticSearch Application Load Balancer)**: The address can be added as a CNAME record for a custom domain if desired
2. **KibanaALB (Kibana Application Load Balancer)**: The address can be added as a CNAME record for a custom domain if desired



## DEPLOYING OPENVPN

This step will deploy an OpenVPN server and publish the OpenVPN configuration file in S3 for clients to connect with. Configuration parameters for the OpenVPN Cloud Formation template are:

1. **OpenVPNPort:** This is the port you want OpenVPN to be running. Default port is 1194.
2. **OpenVPNProtocol:** Specify the protocol for the above port that OpenVPN will listen on. Default is UDP.
3. **PublicSubnet:** Specify the Public Subnet deployed in step 6.3.1
4. **RouteTableId:** This is the route table that was created as part of step 6.3.1
5. **VpcId:** This is the AWS Workspaces VPC that was deployed as part of step 6.3.1

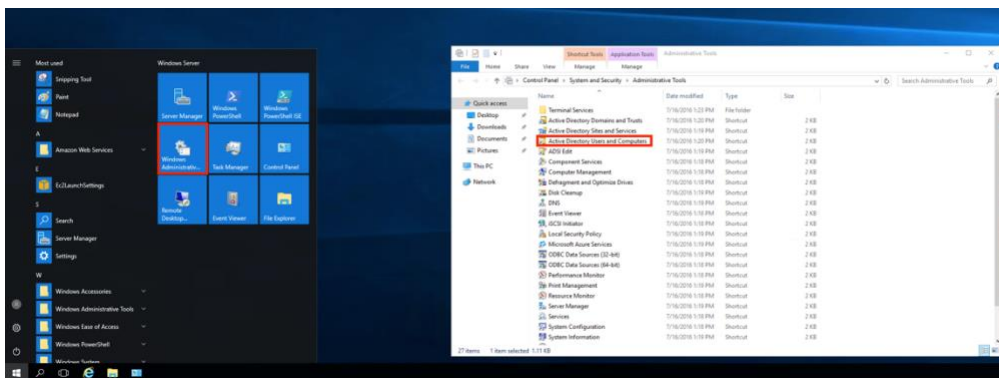
As part of the deployment of the OpenVPN service an S3 bucket is also provisioned for OpenVPN logs and also this is where the OpenVPN profile is saved too. Once the VPN server is provisioned the profile and logs are available in the following S3 location  
<CloudFormationDeploymentName>-sandboxvpnbucket-<random chars> under the client folder.

## 5. ADMINISTRATION

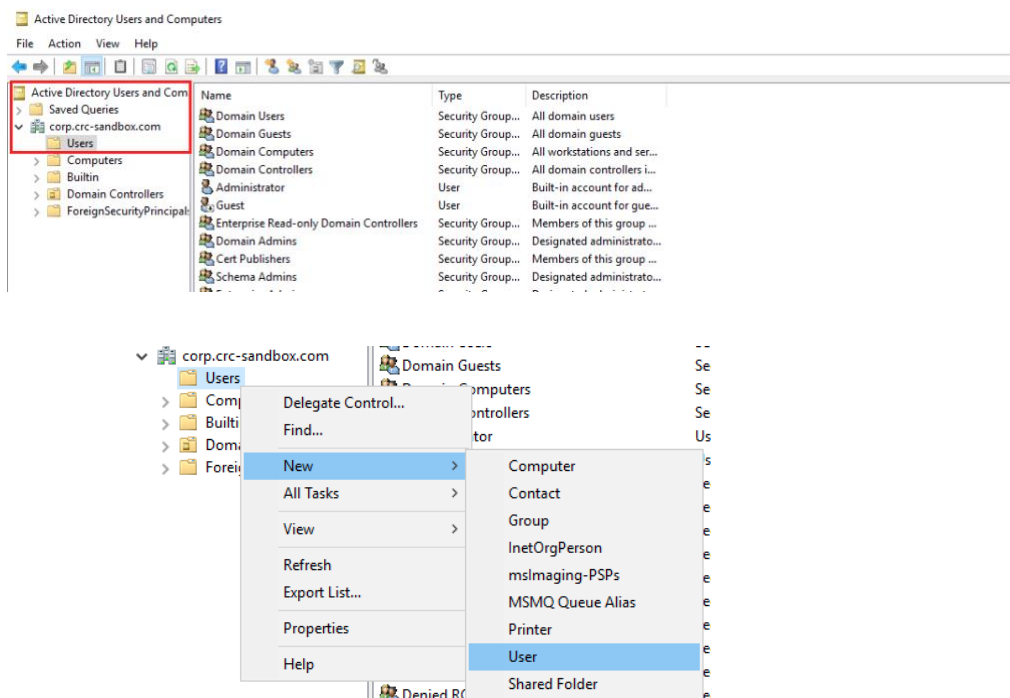
### 5.1 ADDING USERS

Users can be added to Amazon Directory Service using the Administrative VM that was provisioned in section 6.3.2. This VM comes with Remote Server Administrative Tools (RSAT) installed specifically for Active Directory and DNS roles. The following steps outline how to access the Active Directory Users and Computers Snap-in and add a new user:

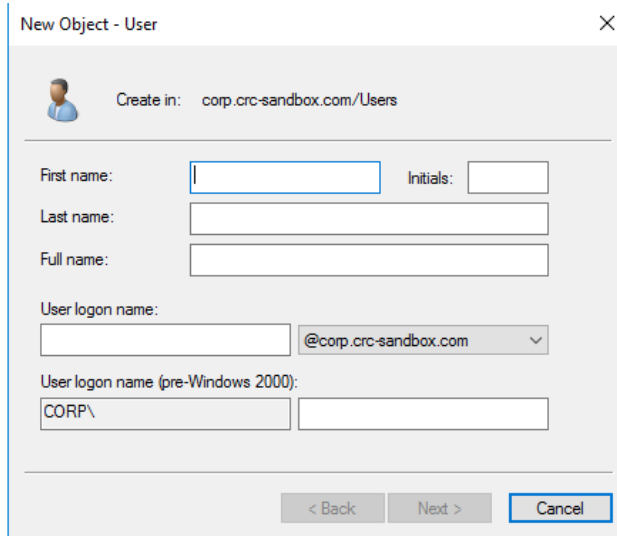
1. Click the Start Menu->Windows Administrative Tools->Active Directory Users and Computers:
2. In the subsequent window that opens navigate to the Users folder:



3. Right-Click Users->New->User



4. On the form that appears enter the user's details and click Next



New Object - User

Create in: corp.crc-sandbox.com/Users

First name:  Initials:

Last name:

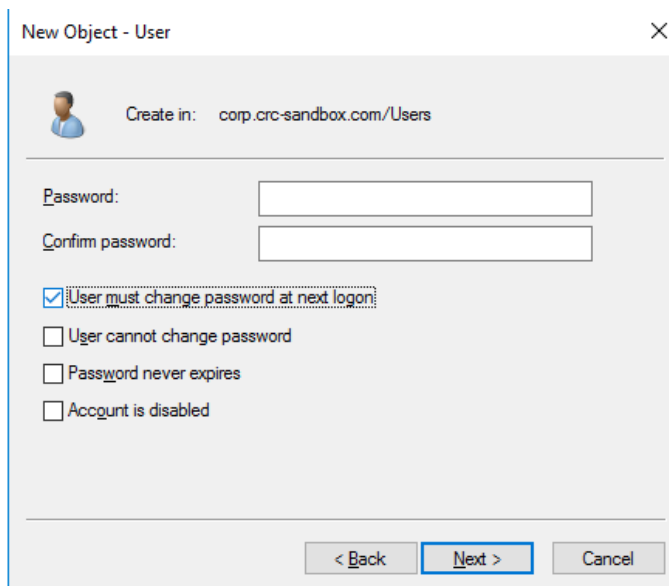
Full name:

User logon name:  @corp.crc-sandbox.com

User logon name (pre-Windows 2000): CORP\

< Back Next > Cancel

5. Set the user's password in the next window. It is recommended that the checkbox "user must change password at next logon" is checked.



New Object - User

Create in: corp.crc-sandbox.com/Users

Password:

Confirm password:

☒ User must change password at next logon

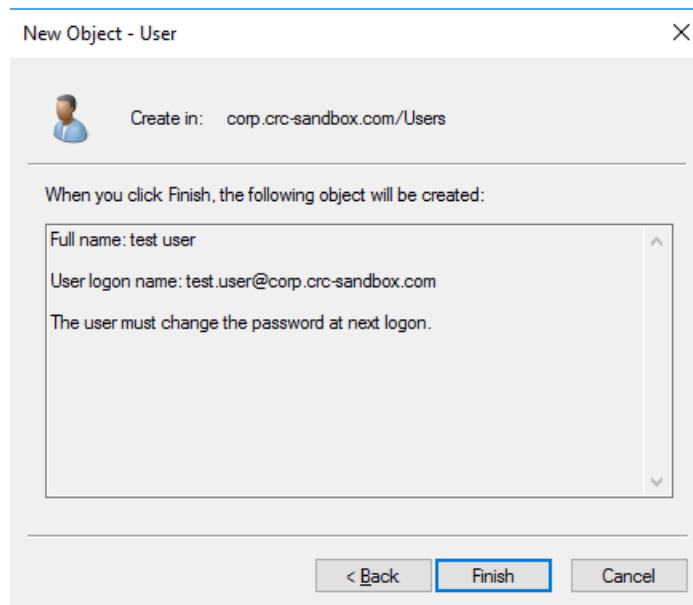
☐ User cannot change password

☐ Password never expires

☐ Account is disabled

< Back Next > Cancel

6. Click Next and then Finish. The user is added to Active Directory. The user is now able to have an AWS Workspace assigned to them.



## 5.2 ADDING RESEARCH VXDS

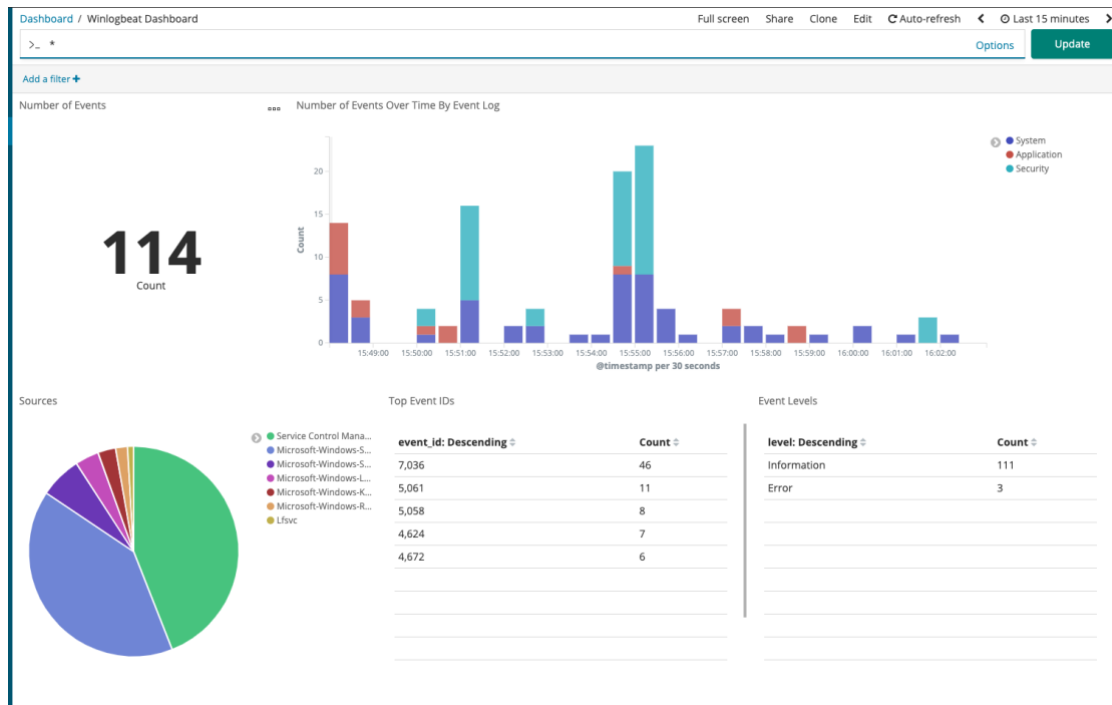
With the user added to Active Directory we are now able to assign them to an AWS Workspace. To deploy an AWS Workspace for a specific user run `awsworkspace-deploy.sh` located in `/workspaces/3-AWSWorkspaceDeploy/` with the username as a command line argument. For example if the username is `john.citizen` simply run `sh awsworkspace-deploy.sh "john.citizen"`

## 5.3 ADDING DATA SETS

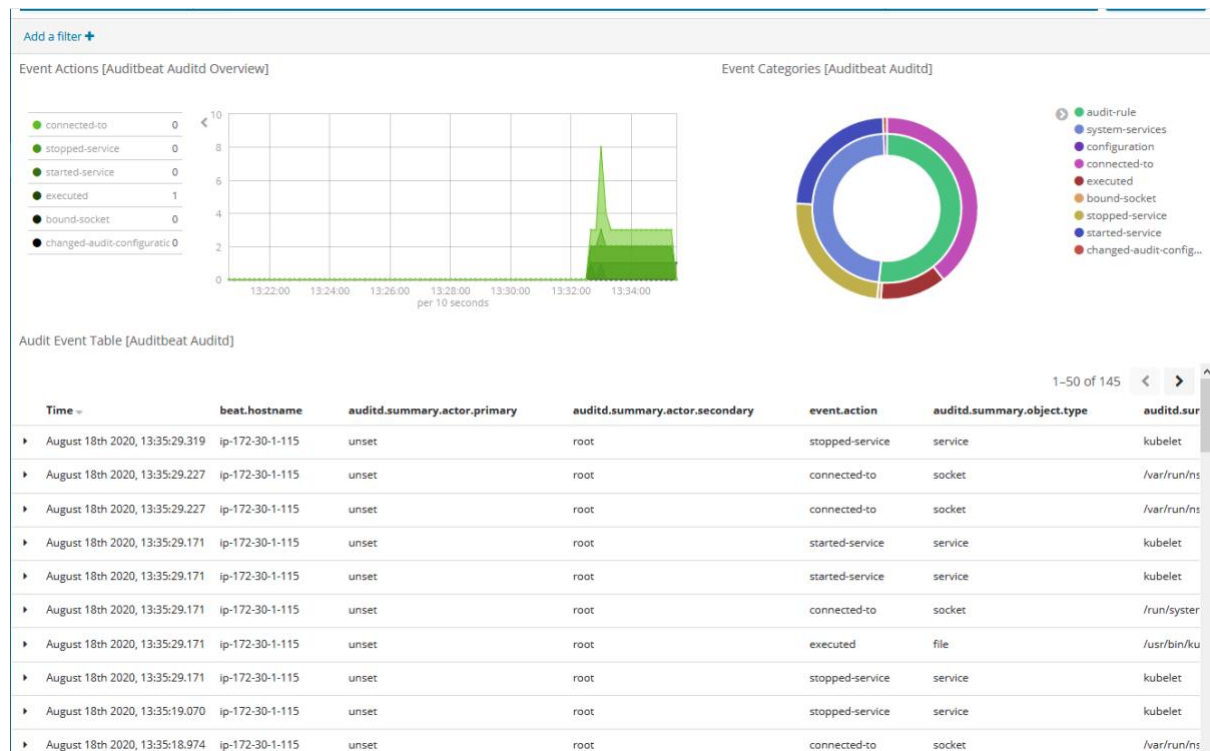
Data sets can be added to the S3 Bucket deployed in section 6.3.5 using either the AWS Web Console, AWS CLI (<https://docs.aws.amazon.com/cli/latest/reference/s3/>) or various 3<sup>rd</sup> Party GUI tools.

## 5.4 ELK EXAMPLES - TRACKING UNDESIRABLE ACTIVITY

Tracking Windows Event Viewer Logs by ID (<https://www.myeventlog.com/>)



## Tracking Unix Audit logs and auditing of commands run by users.



## 6. EXTENSIONS

---

### 6.1 ALLOWING ACCESS TO MULTIPLE VMS FROM THE VXD

To facilitate access to other VMs from AWS Workspaces ensure the VMs are deployed into one of the private subnets in the Workspaces VPC. Alternatively, if a new VPC was to be created and VMs are required to be accessed from it (via Local IP), VPC peering is required. More information on VPC peering can be found here

<https://docs.aws.amazon.com/vpc/latest/peering/what-is-vpc-peering.html>

### 6.2 MFA INTEGRATION WITH AWS WORKSPACES AND OPENVPN

To further secure the virtual desktop environment and VPN it is recommended adopting a MFA service such as Azure AD MFA, Okta, PingIdentity, Duo or something similar that allows users to authenticate with a second factor to prevent compromised accounts from gaining access to the environment. AWS Workspaces has support for MFA providers and the support for MFA and OpenVPN is available as well.

### 6.3 CONFIGURATION OF REMOTE SUPPORT SERVICE FOR AWS WORKSPACE

Part of the default configuration of AWS Workspace is a security policy that doesn't allow AWS Workspaces to communicate with each other. Further to this Windows doesn't allow more than one active session to be logged into the machine. As a result supporting end users who have issues with their Workspace may be challenging. An extension to this solution would be to configure a remote assistance solution which allows screensharing with a support agent, eg. TeamViewer, VNC, etc. This will allow support to screenshare with the affected user and provide assistance where needed.

### 6.4 OTHER

The AWS Workspaces in this example can be strengthened to make data copying even harder. The user may want to consider locking the system down further to make data movement even harder. This is something to be cautious about as it can restrict the ability of the researcher to do their job.

Some example are:

- Lock down administrator privileges
- Install web proxies to restrict use of services like Gmail and other Google tools, Dropbox, etc.
- Install systems to record screen activity. Some people have implemented systems to OCR screen captures to further detect undesirable activity.