UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Department of Computer Science

COS 301 - Software Engineering

Nimbus
Functional Requirements

Authors:	*Student number:*
Jedd Schneier	u13133064
Daniel King	u13307607
Muller Potgieter	u12003672

October 23, 2016

Software Requirements Specification and Technology Neutral Process Design

Nimbus AWS Network Visualiser/Main Project

Version: Version 1.0 Beta For further references see gitHub. October 23, 2016

Contents

1 Functional requirements

1.1 Introduction

The Nimbus Amazon Web Services (AWS) network visualiser will be used to visualise a user's network within the AWS network, through their browser. The purpose of this document is to identify and explain all possible use cases associated with the visualiser and to show how the functional aspects of the visualiser interact with each other.

1.2 Use case prioritiation

Critical

- Log in/Log out
- Scan network
- Visualise network
- Get Node Connections
- Get Node information

Important

- Stop scan
- Resume scan
- Scan Up
- Scan Region
- Scan From
- Scan Instances

Nice-To-Have

- Load scan from local .json
- Save scan to local .json

1.3 Use case/Service contracts

Use Case	Pre Condition	Post Condition	Description
Log in/log out	The visualiser must be connected to the internet in order to verify the login information. Only a registered AWS user with a valid key and secret key may log into the system. Once the user is logged in he/she can then use the logout functionality to log out.	The user is logged in now and may begin making use of the visualisation server.	This use case provides a method for logging in to view a hierarchical representation of their network and log out once done.
Scan network	The visualiser must be connected to the internet in order to load the information from the server. The user must have instances in their network, that the server may scan.	The server continuously sends information of the instances to the browser, which are then stored by the browser.	This use case provides a method for loading the network representation from the AWS network and storing it on a browser.
Visualise Network	The visualiser must be connected to the internet in order to load the information from the server. The user must have instances in their network, that the server may scan. The browser must have a feed from the server, that contain instance information.	As new node information is loaded into the browser, the browser will sequentially display them in a hierarchy.	This use case provides a method for reading information the browser has received and visualising it.
Get Node Connections	The visualiser must be connected to the internet in order to load the information from the server. The user must have instances in their network that the server may scan.	The relationships between nodes are logged and added to the buffer.	This use case provides a method for logging the relationships of instances.

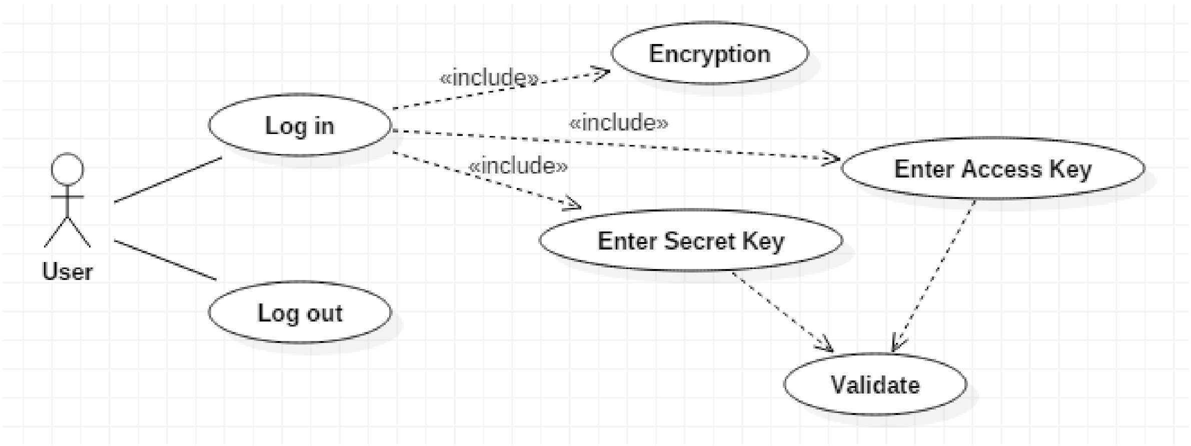
Get Node Infromation	The visualiser must be connected to the internet in order to load the information from the server. The user must have instances in their network, that the server may scan. The browser must have a feed from the server, that contain instance information.	Information relating to the specified instance is displayed.	This use case provides a method for easily viewing infromation related to specific instances/nodes.
Stop scan	The scan must be active.	The scan's execution is temporarily halted.	This use case provides a method for temporarily halting an active scan.
Resume scan	An active scan must have been stopped.	The scan resumes its execution.	This use case provides a method for resuming a previously halted scan.
Scan Up	The scan must be active.	The direction of the scan is altered	This use case provides a way of changing the direction of the scan.
Scan Region	The scan must be active.	The scan refocuses and only scans instances that fall below a certain region.	This use case provides a way to only scan instances that belong to a specific AWS region.
Scan From	The scan must be active. The user has provided a valid name of an instance.	The scan starts at the provided instance and begins scanning from the specified point.	This use case provides a way of specifying where the scan should begin.
Scan Instances	The scan must be active.	The scan only scans instances.	This use case provides a way to only scan instances.
Load scan from local .json	A .json file with the correct formatting must be stored on the local device.	The browser processes the information on the .json and visualises it, as it would with a normal scan.	This case provides a way load network visualisations, without the need to scan it from the server.

Save scan to local .json	The scan must be active/ finished.	A window appears that will save a .json file to the local device. The file is named with a time stamp, in order to avoid issues if multiple files are saved	This case provides a way to save a representation of the current scan, that can be loaded at a later date.

1.4 Required functionality

- **Log in/log out**

  Log in allows the user to log in using a valid AWS Access Key and Secret Key. Their details are encrypted, to avoid security leaks. Only once a user is logged in, may they use the system. Once they have finished they may log out. Doing so will clear the credentials used, to ensure privacy.
  Description
  This function validates clients' credentials to allow them to scan an accounts network.
  Inputs
  Clients access key and clients private key.
  Processing
  The access key and private key is passed to a Credentials object. The credentials are validated by trying to connect to an account with those keys.
  Outputs
  If validation successful:
  Access to the visualization interface.
  If invalid:
  Error message and retry prompt.



- **Scan network**

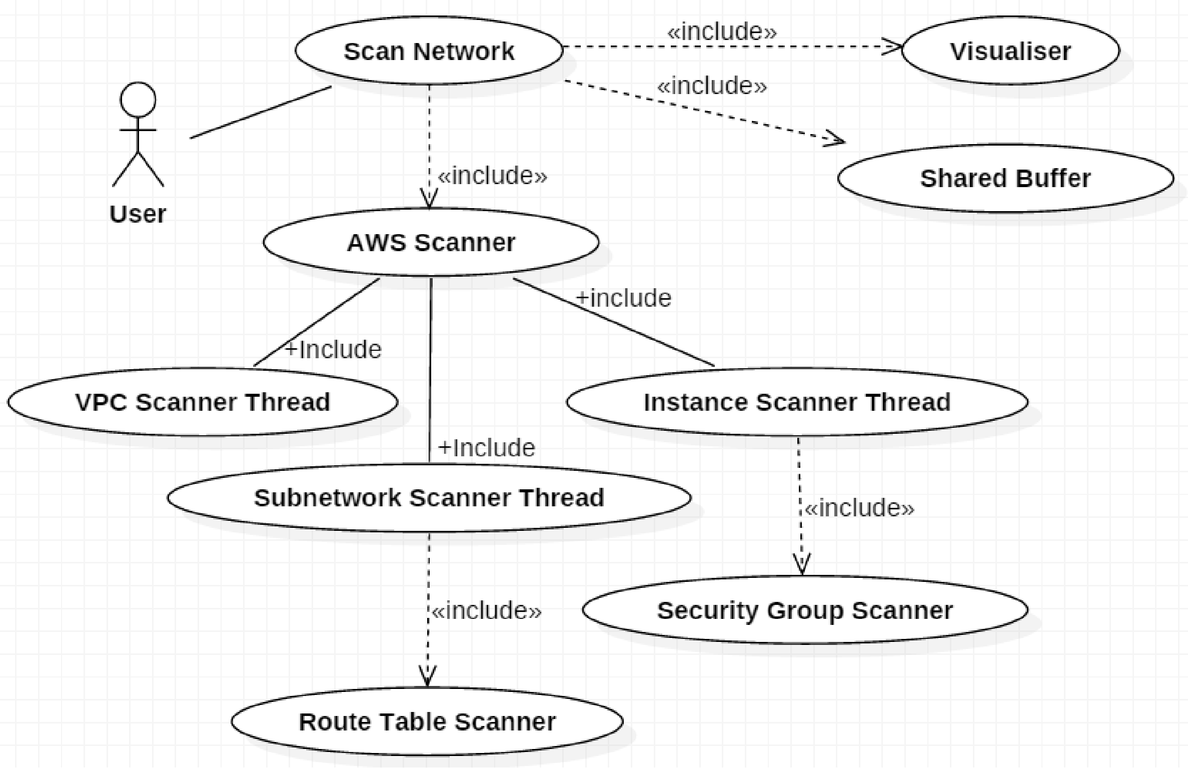
  When the scan is initialised, the AWS scanner is activated. It will create a number of threads that will scan the various parts of the network simultaneously, greatly increasing the scalability and performance. As the scans occur, the results are added to the Shared Buffer, that the Visualiser reads from and draws the nodes as they appear.

  Description
  This function launches a scan of the entire network

  Inputs
  Previously validated credentials

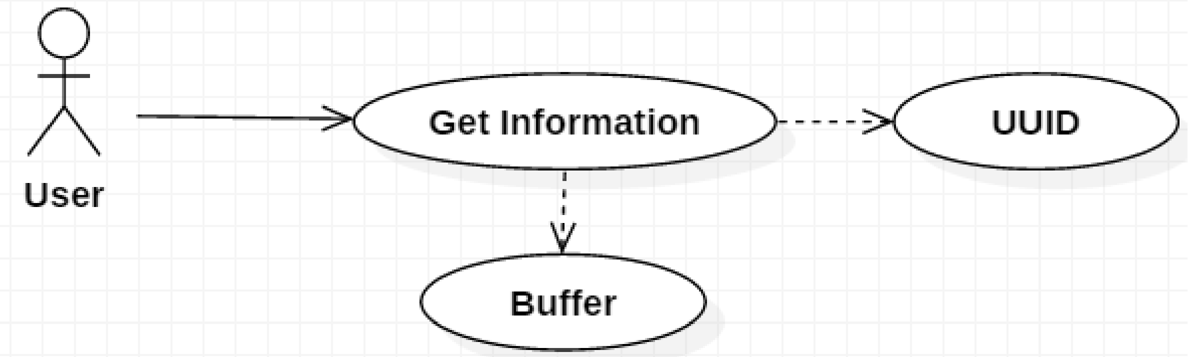
  Processing
  A number of subscanners are launched. They add whatthey find to the buffer. The buffer constructs the tree and the visualizaer fetches it.

  Outputs
  A dynamically growing tree that is rendered to user.



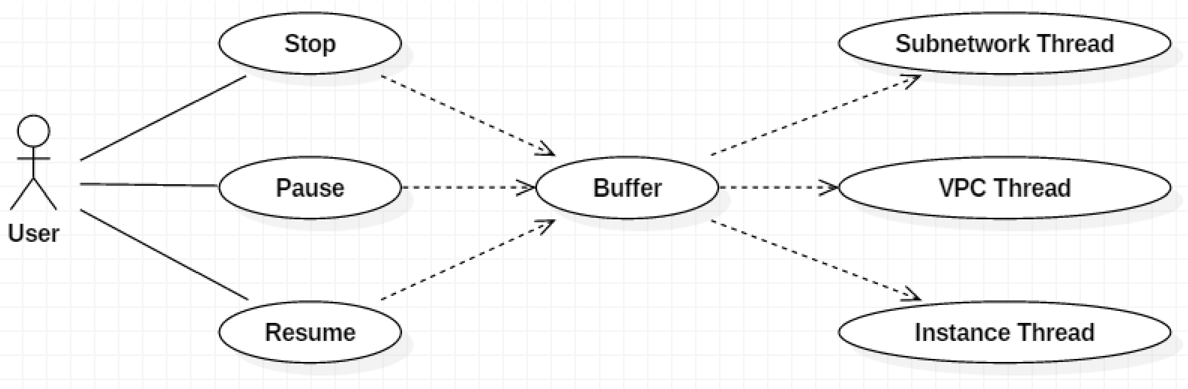
- **Get Node Information**

  If a request is sent for a node's information, the shared buffer is queried [using the instances unique ID (UUID)] and the results are sentto the visualiser.
  Description
  This function requests inforamtion for a specific node
  Inputs
  The nodes uuid.
  Processing
  The buffer looks for the matching information in its hashmap.
  Outputs
  A set of information about the node.



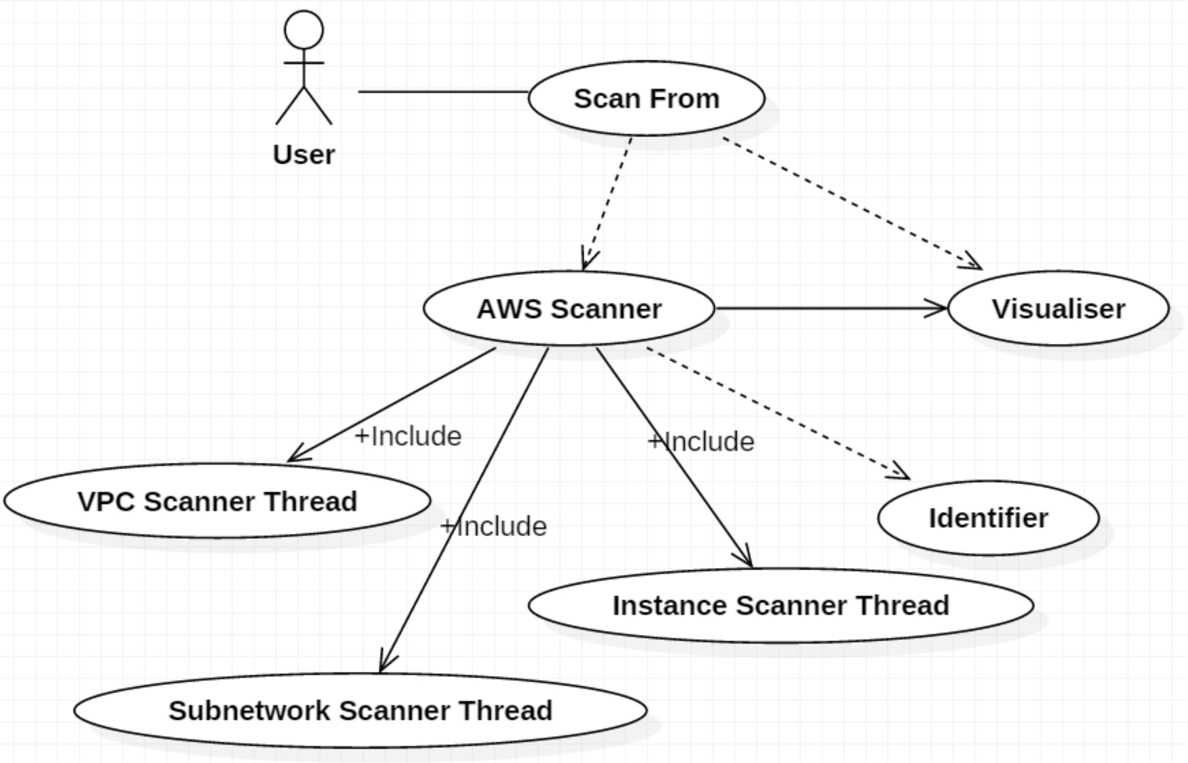
- **Stop/Pause/Resume Scan**

  If a stop/pause/resume request is sent, the buffer will forward the request to the threads.



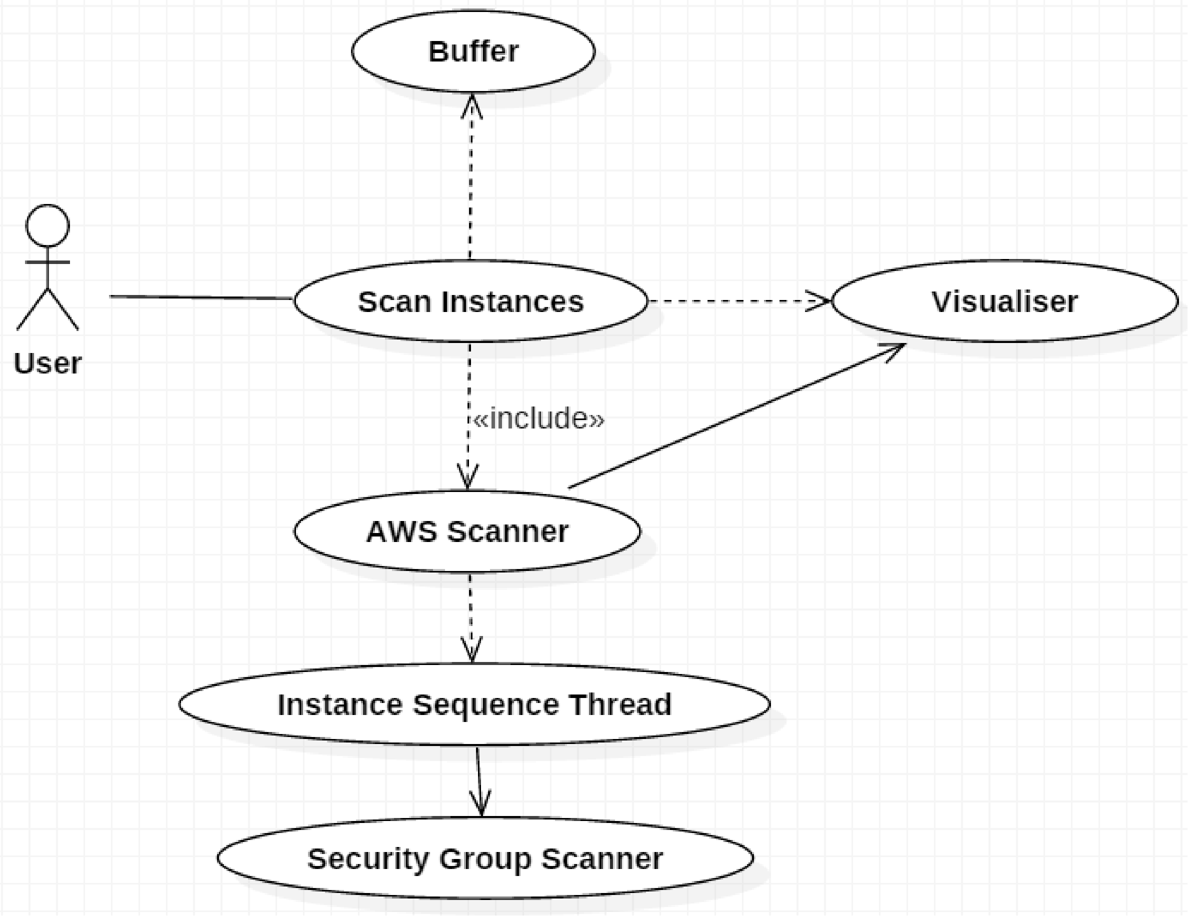
- **Scan From**

  Description
  This function launches a scan of the network from specific uuid.
  Inputs
  Uuid of specific node
  Processing
  The scanner controlelr launches the specific threaded scanner only and it follows the process of scan network, but on its own.
  Outputs
  The network scanned from that spceifed node.



- **Scan Instances**

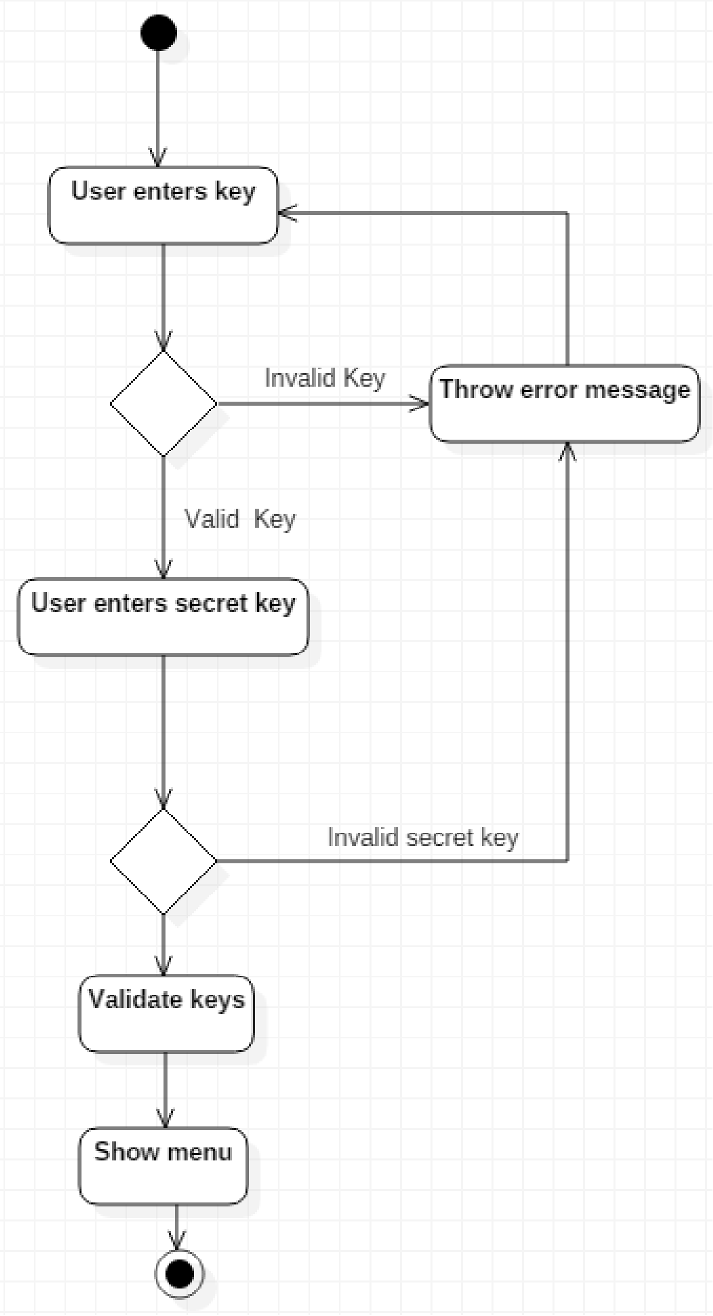
  Description
  This functionscans all isntances registered t othe user.
  Inputs
  Nothing
  Processing
  The scanner controlelr launches an instance threaded scanner for each region only which follow the process of scan network.
  Outputs
  All instances scanned and rendered.



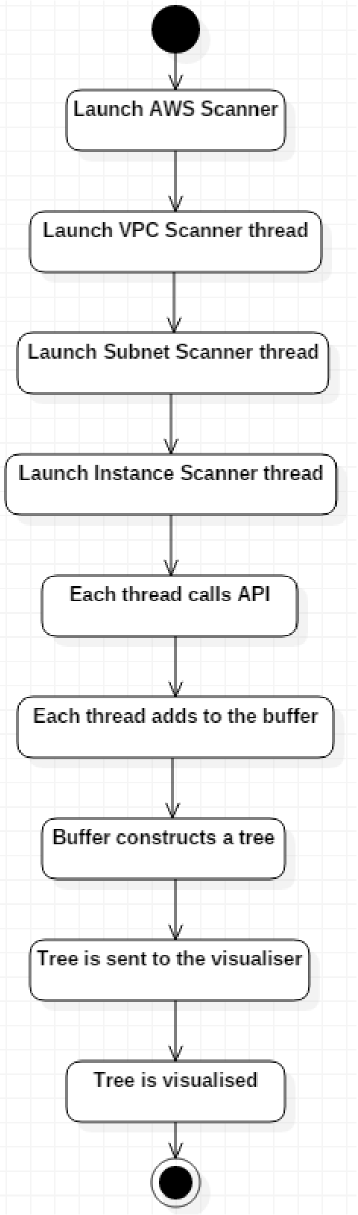
1.5 Process specification

The processes followed when using some of the more important functions of the system are displayed below:

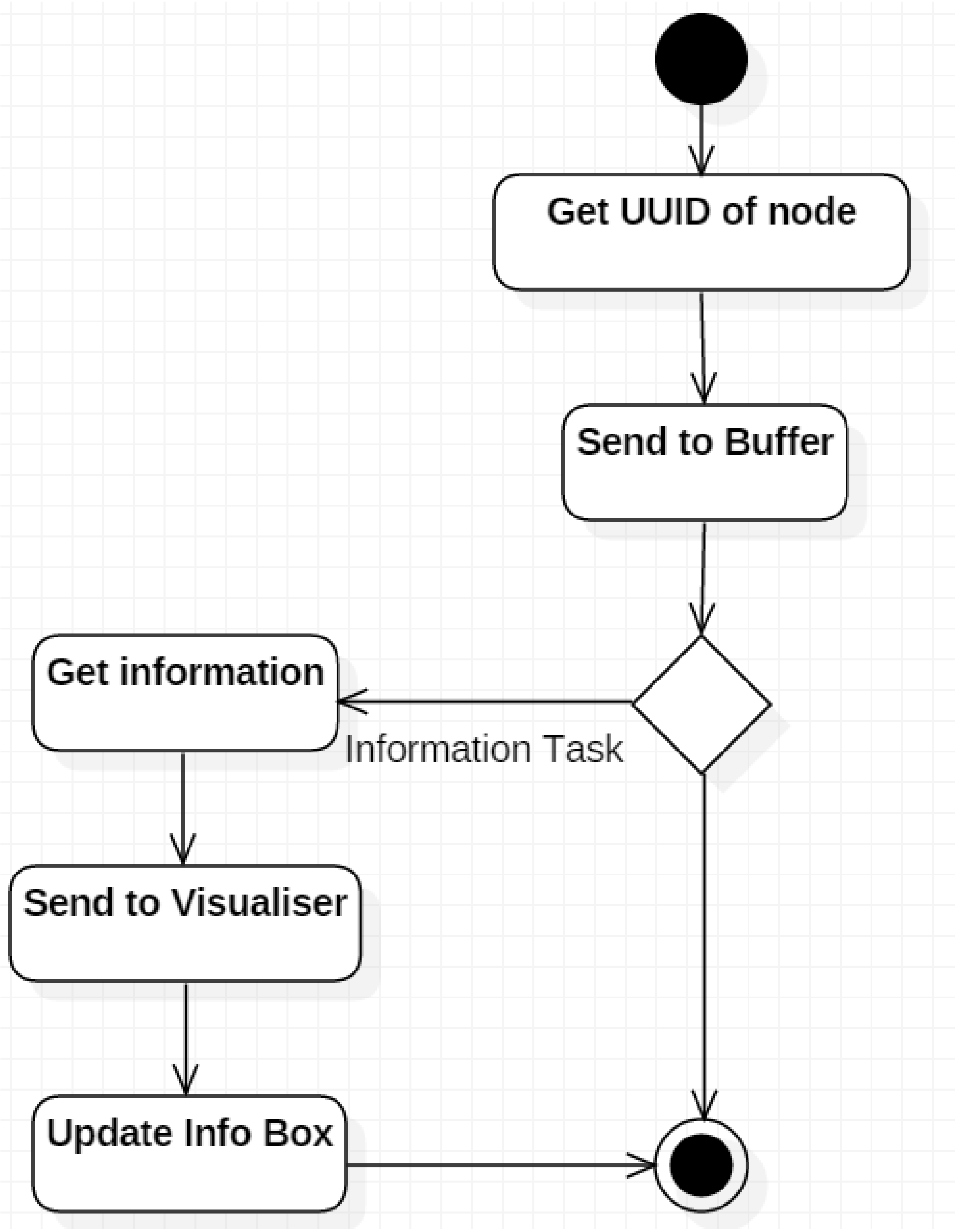
- Log in/ log out



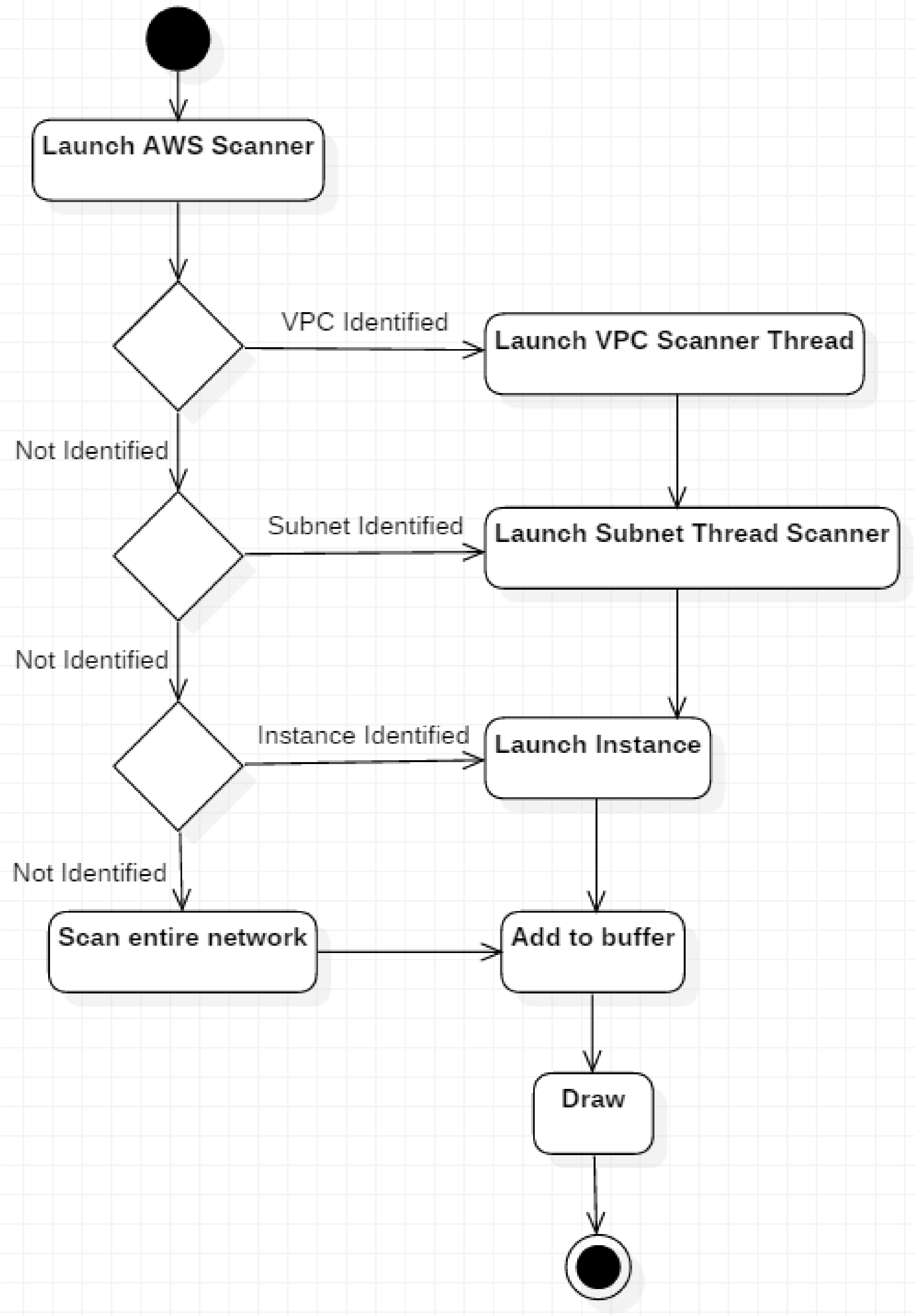
- Scan network


Launch AWS Scanner
Launch VPC Scanner thread
Launch Subnet Scanner thread
Launch Instance Scanner thread
Each thread calls API
Each thread adds to the buffer
Buffer constructs a tree
Tree is sent to the visualiser
Tree is visualised

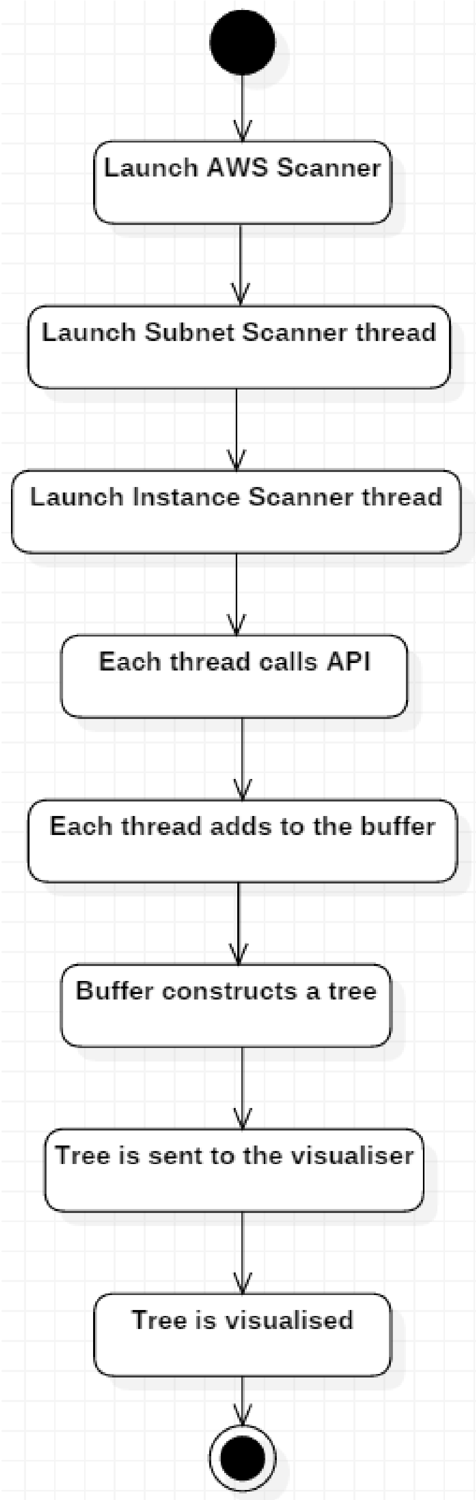
- Get Node Information



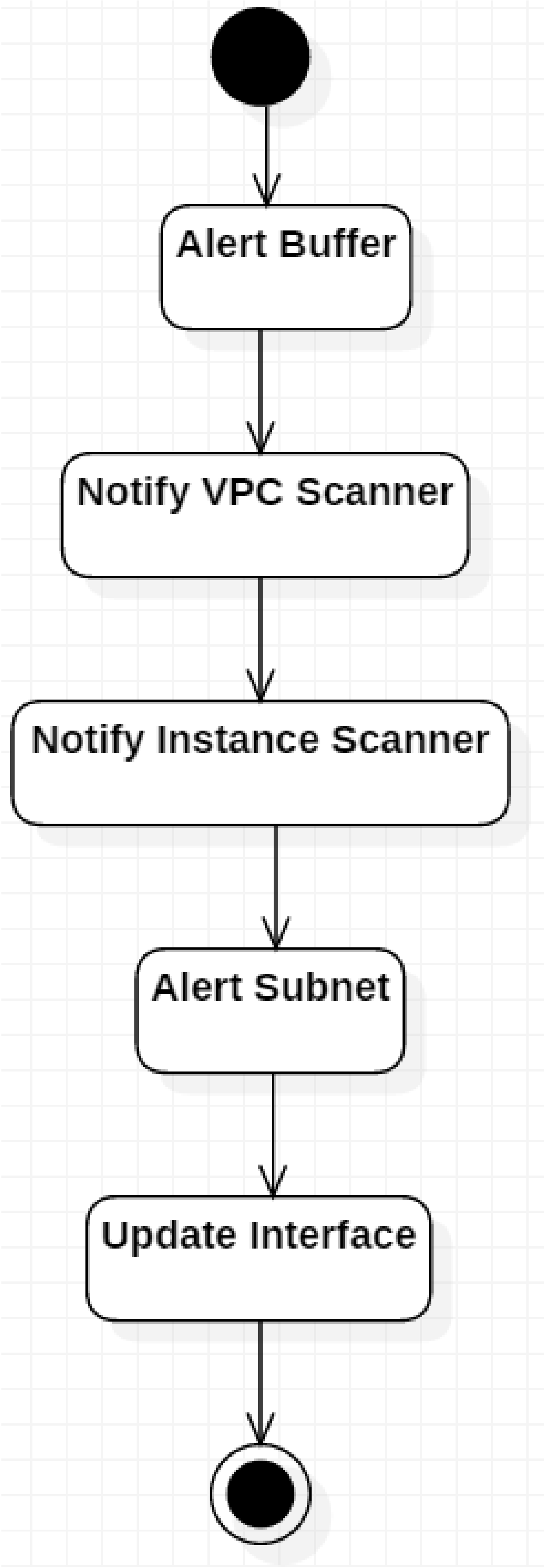
- Scan From


Launch AWS Scanner
VPC Identified
Launch VPC Scanner Thread
Not Identified
Subnet Identified
Launch Subnet Thread Scanner
Not Identified
Instance Identified
Launch Instance
Not Identified
Scan entire network
Add to buffer
Draw

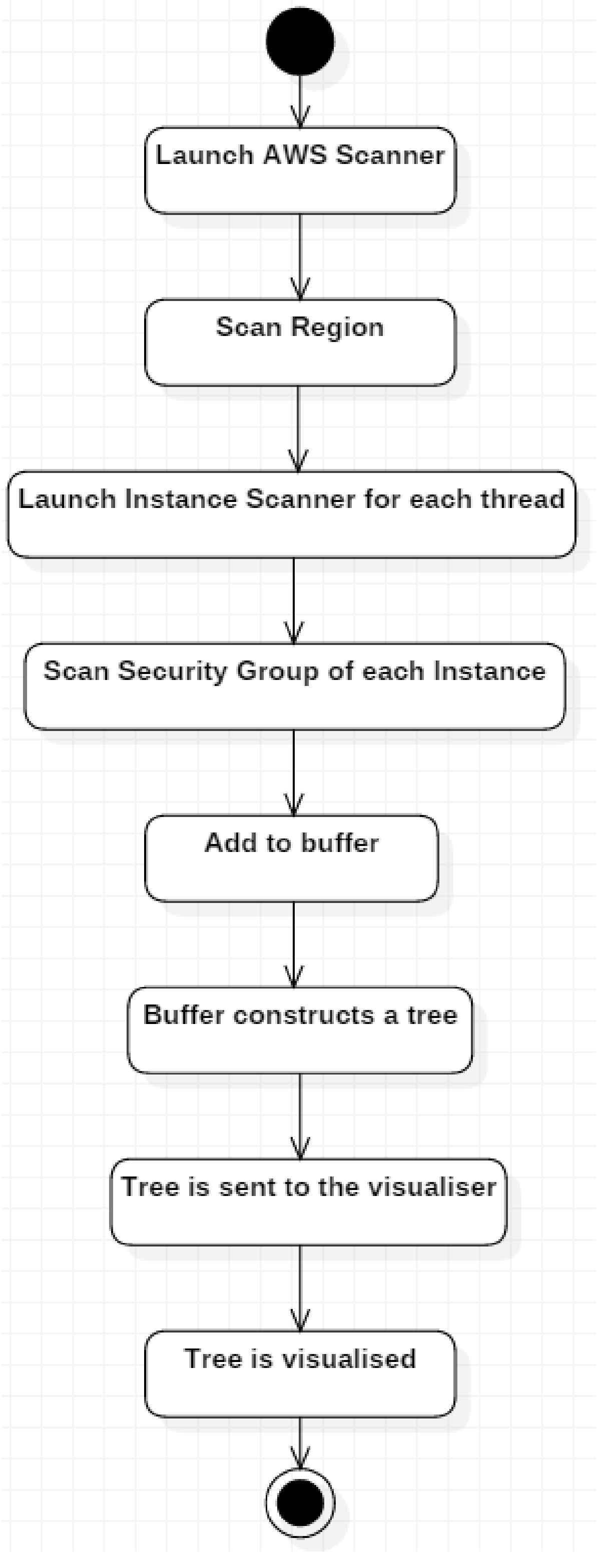
- Scan Region


Launch AWS Scanner
Launch Subnet Scanner thread
Launch Instance Scanner thread
Each thread calls API
Each thread adds to the buffer
Buffer constructs a tree
Tree is sent to the visualiser
Tree is visualised

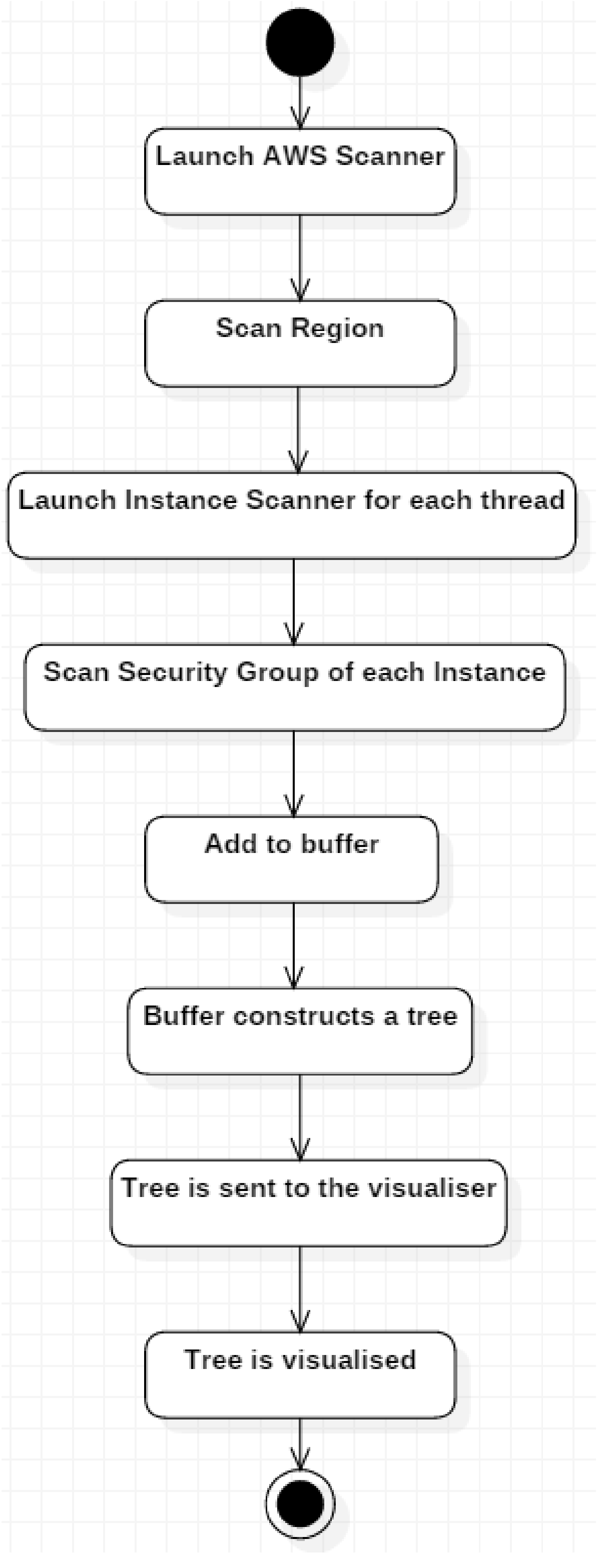
- Change Scanner State



- Scan Instances


Launch AWS Scanner
Scan Region
Launch Instance Scanner for each thread
Scan Security Group of each Instance
Add to buffer
Buffer constructs a tree
Tree is sent to the visualiser
Tree is visualised

1.6 Domain Model


Launch AWS Scanner
Scan Region
Launch Instance Scanner for each thread
Scan Security Group of each Instance
Add to buffer
Buffer constructs a tree
Tree is sent to the visualiser
Tree is visualised