



UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

DEPARTMENT OF COMPUTER SCIENCE

COS 301 - SOFTWARE ENGINEERING



Functional Requirements

Authors:

Jedd Schneier

Daniel King

Muller Potgieter

Student number:

u13133064

u13307607

u12003672

October 22, 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 22, 2016

Contents

1	Functional requirements	3
1.1	Introduction	3
1.2	Use case prioritiation	3
1.3	Use case/Service contracts	4
1.4	Required functionality	6
1.5	Process specification	10
1.6	Domain Model	19

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

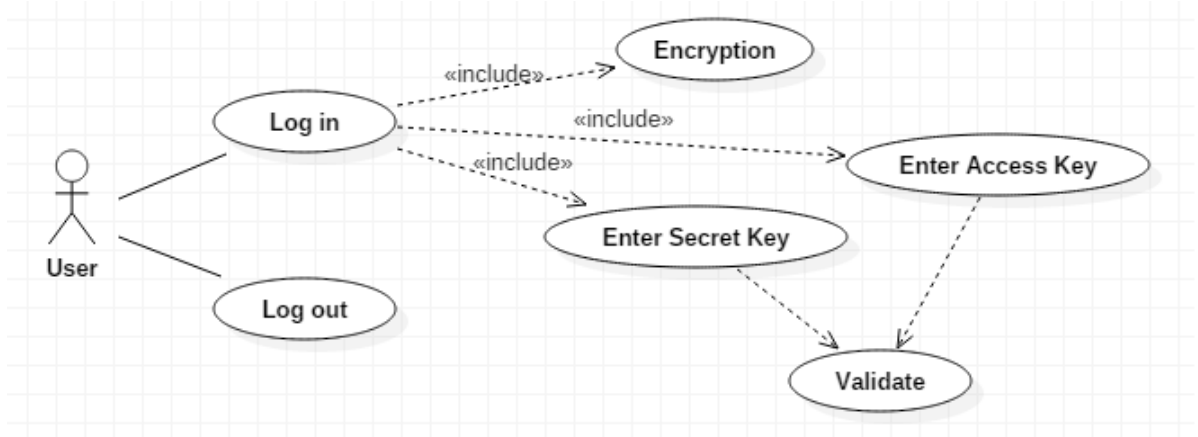
Use Case	Pre Condition	Post Condition	Description
Log in/log out	The visualiser has to be connected to the internet in order to verify the information. Only a registered AWS user with a valid key and secret key may log into the system. Once the user is logged in he/she can then use the logout functionality 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	aaa.	aaa.	aaa.
Get Node Information	aaa.	aaa.	aaa.
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	aaa.	aaa.	aaa.
Scan Instances	aaa.	aaa.	aaa.
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

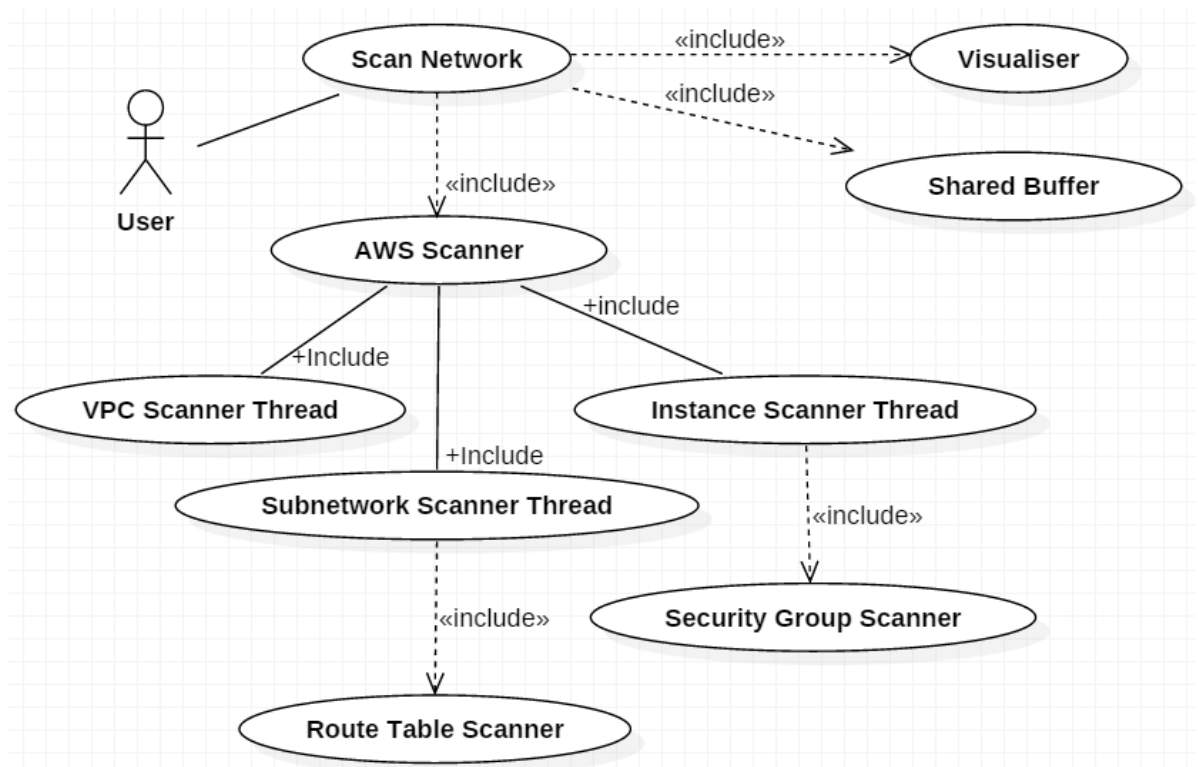
- Log in/log out

Basically just say what this part does.



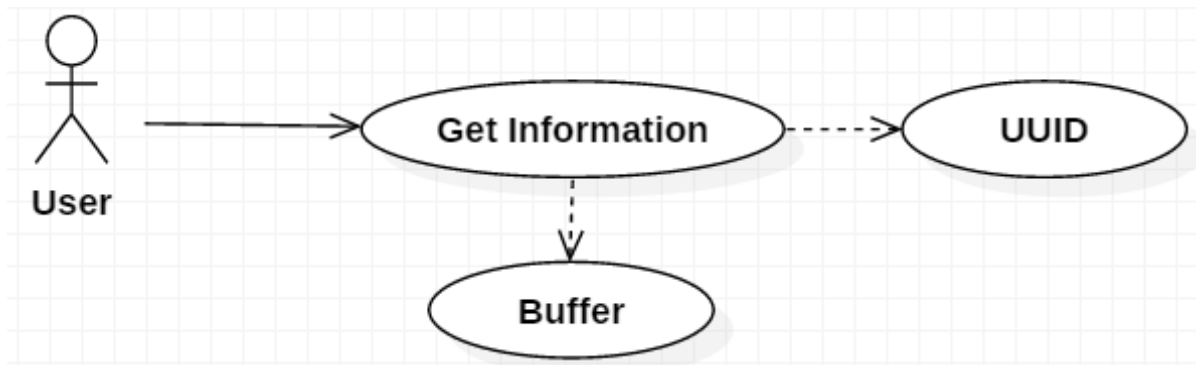
- Scan network

Basically just say what this part does.



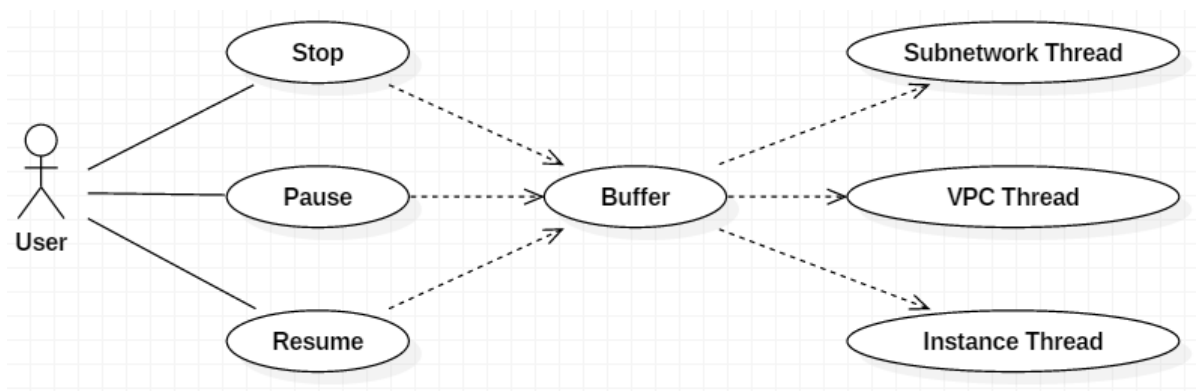
- Get Node Information

Basically just say what this part does.



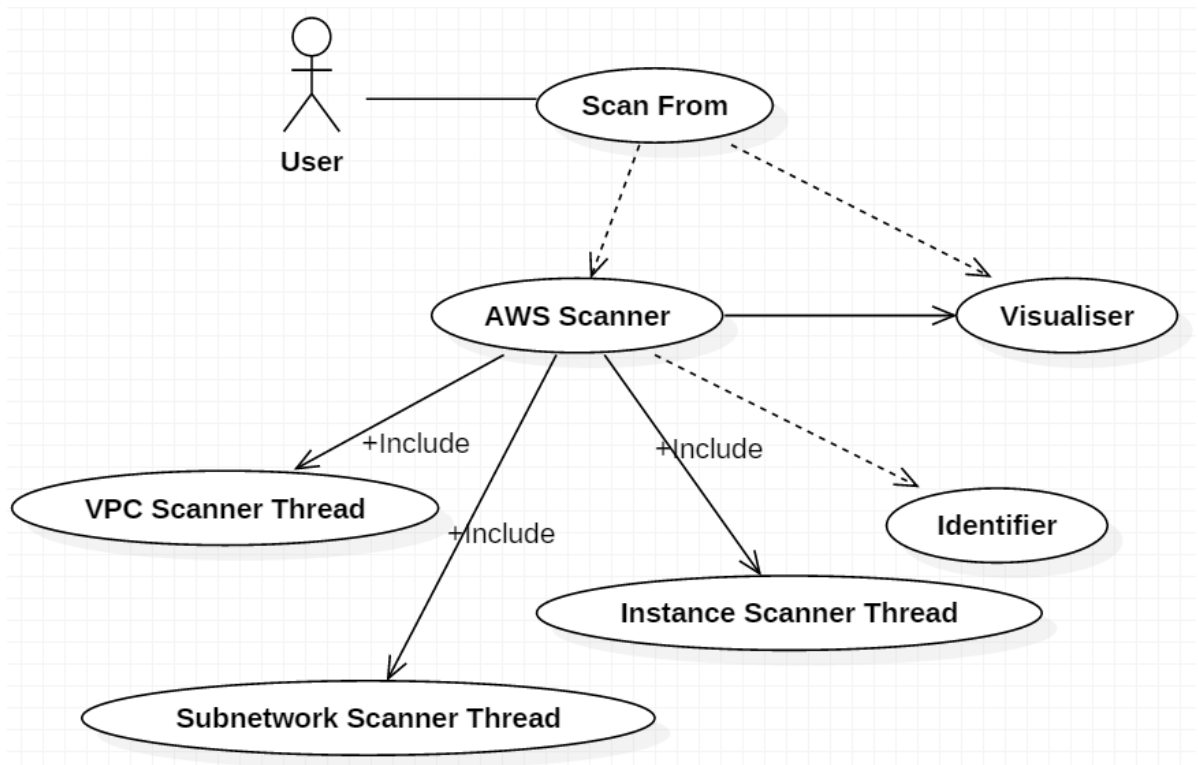
- **Stop/Pause/Resume Scan**

Basically just say what this part does.



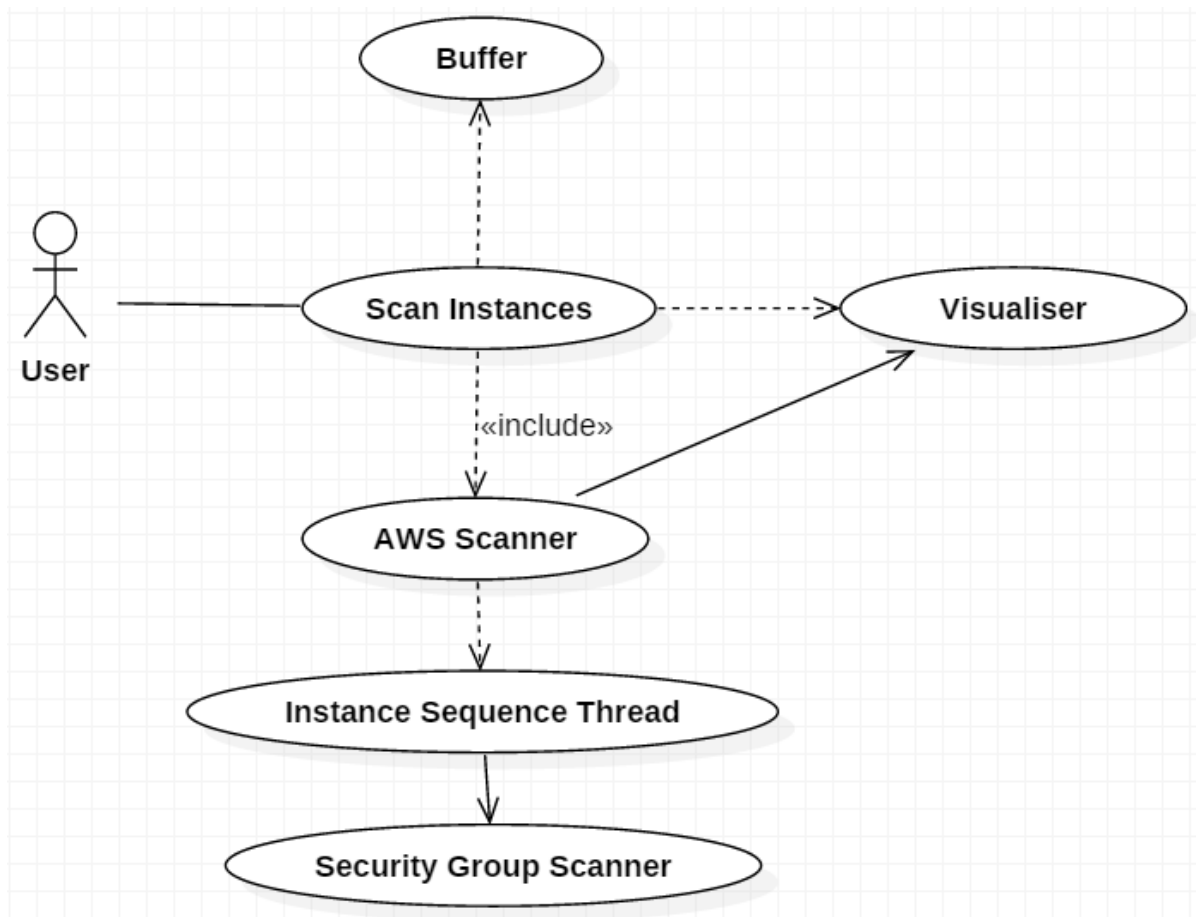
- **Scan From**

Basically just say what this part does.



- **Scan Instances**

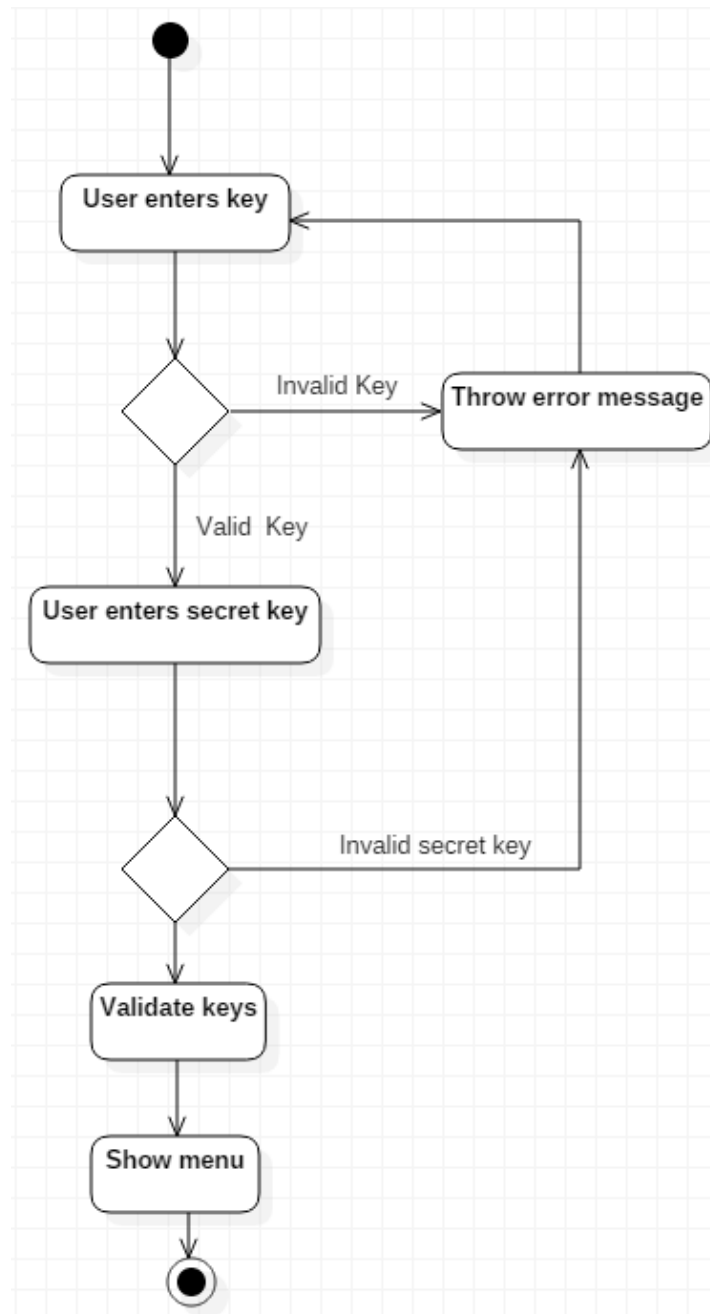
Basically just say what this part does.



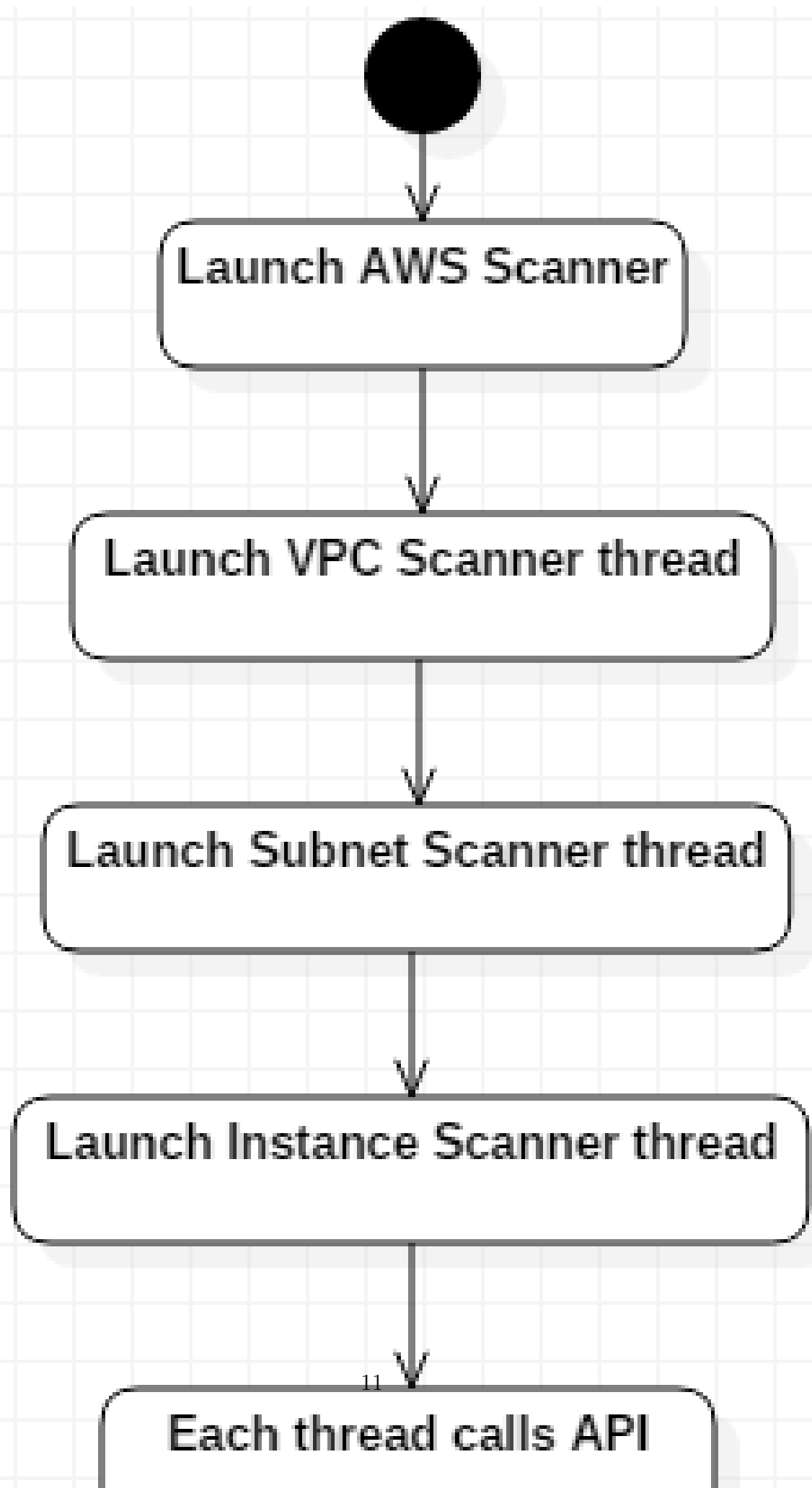
1.5 Process specification

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

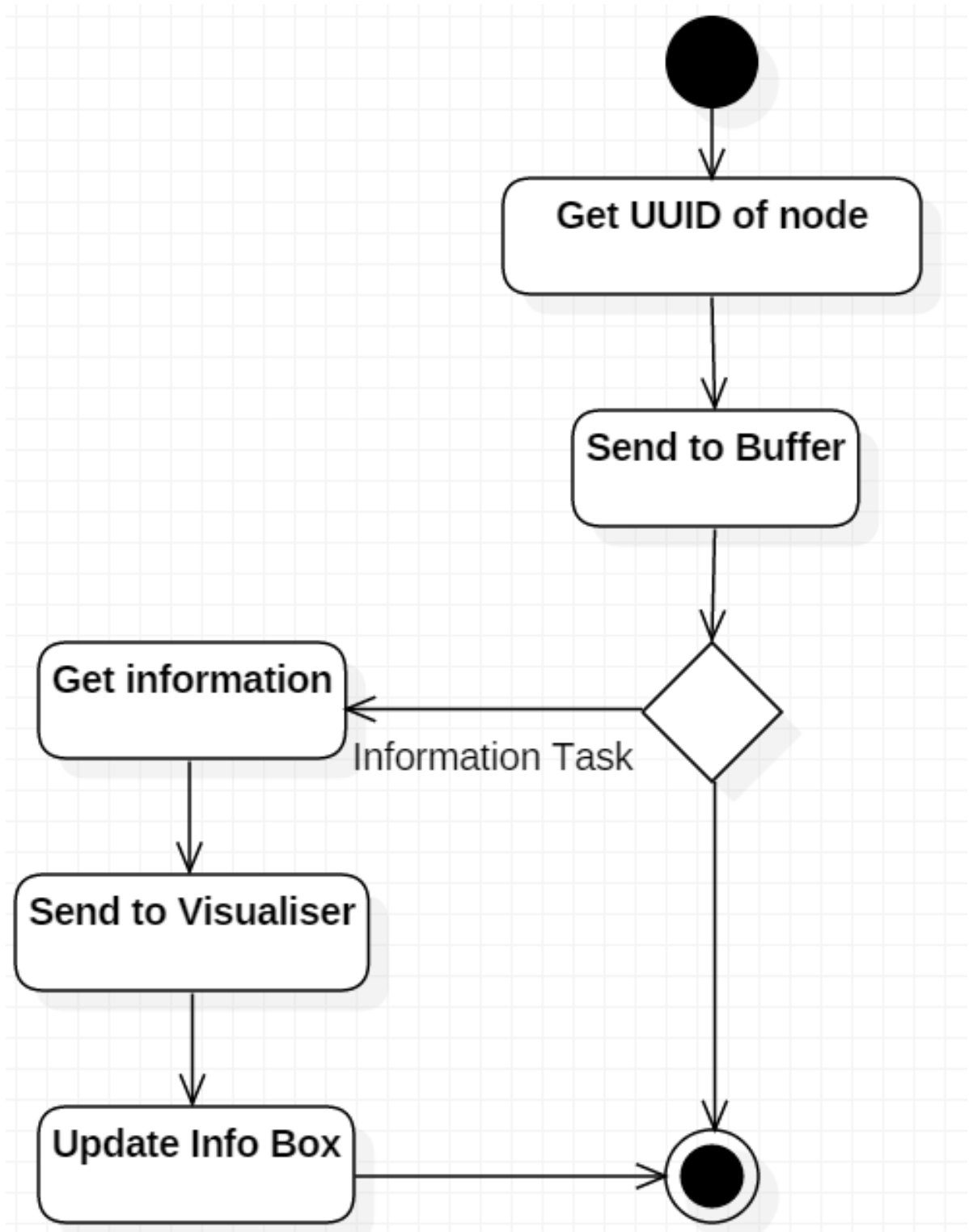
- Log in/ log out



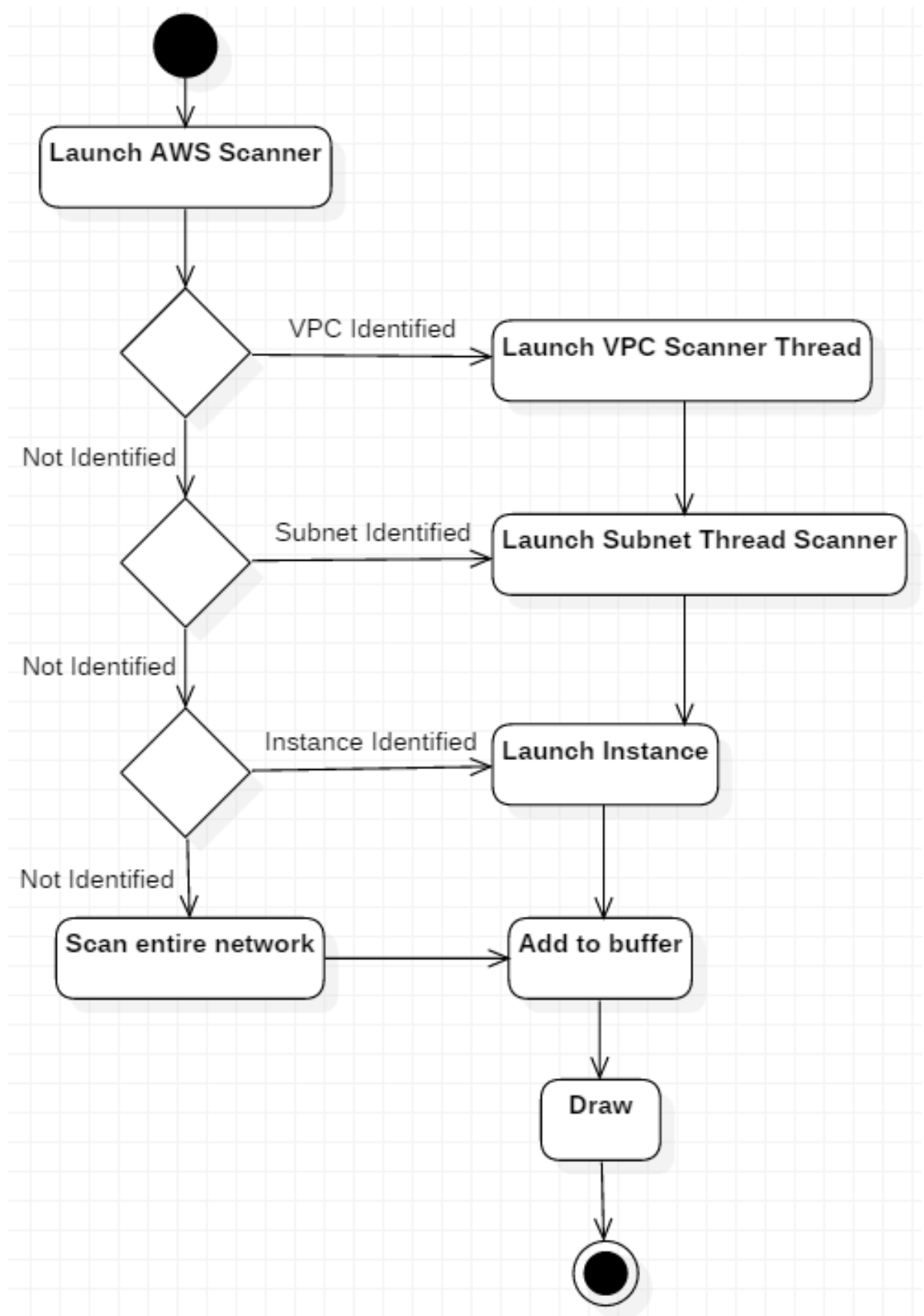
- Scan network



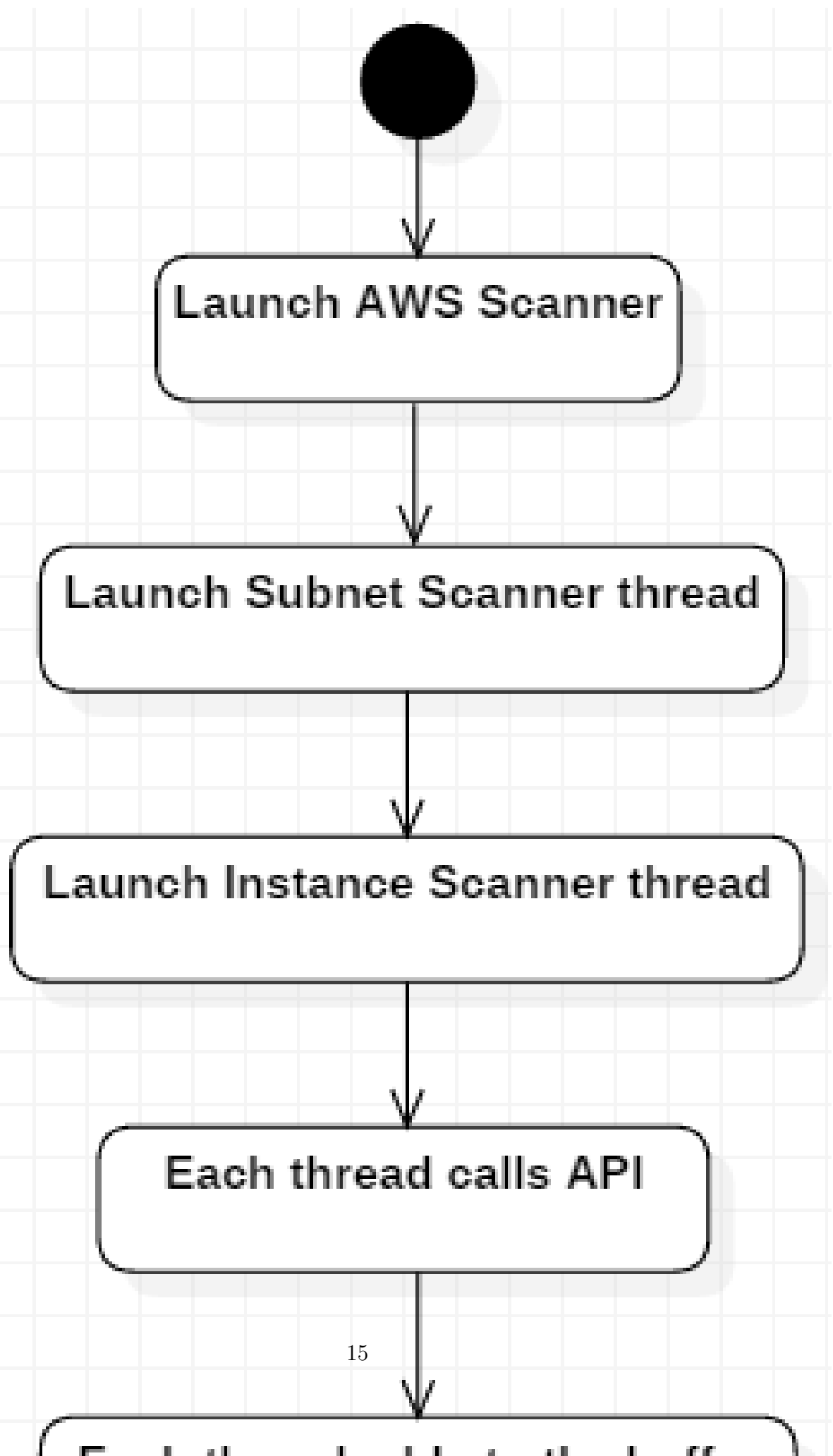
- Get Node Information



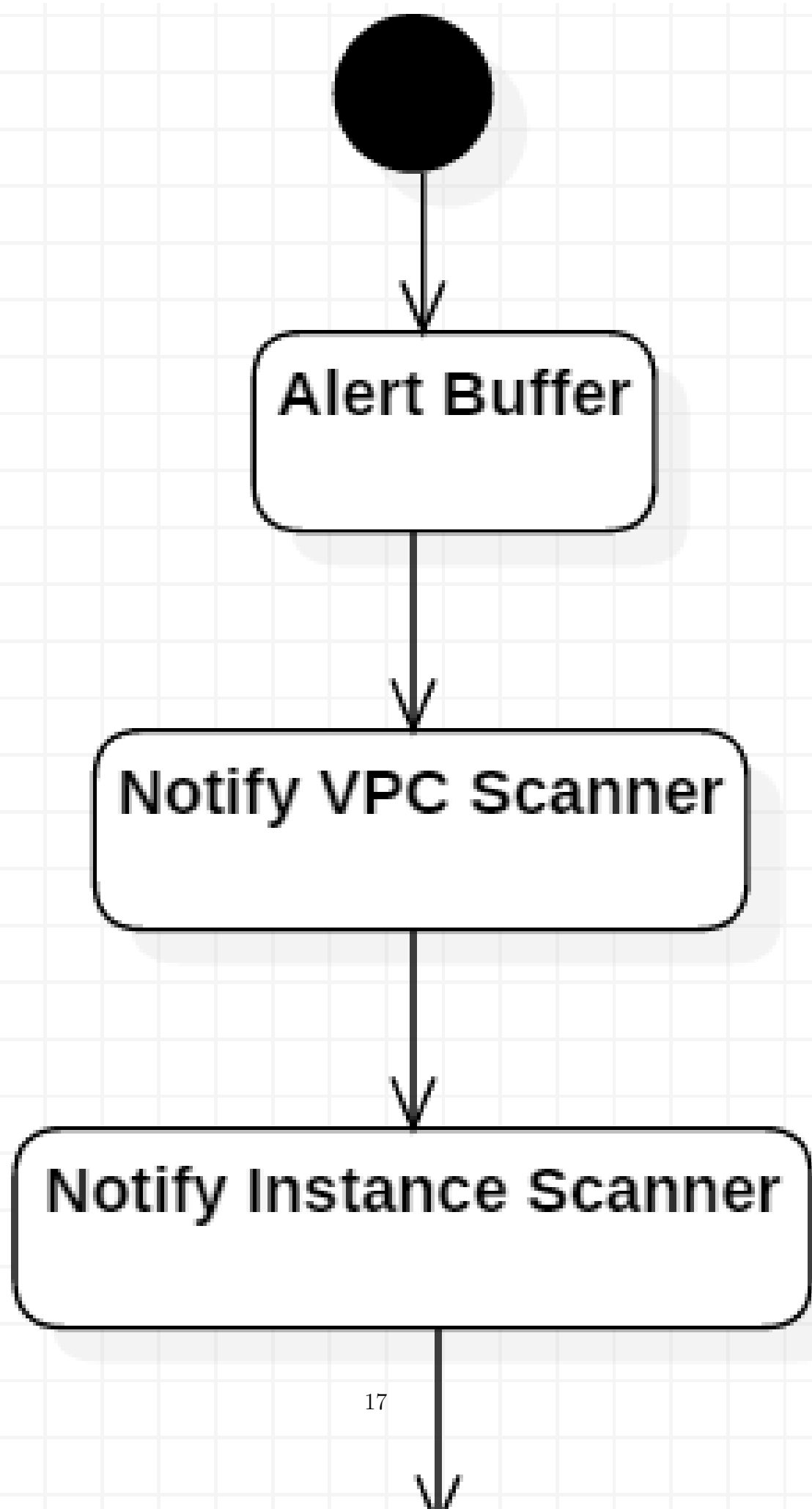
- Scan From



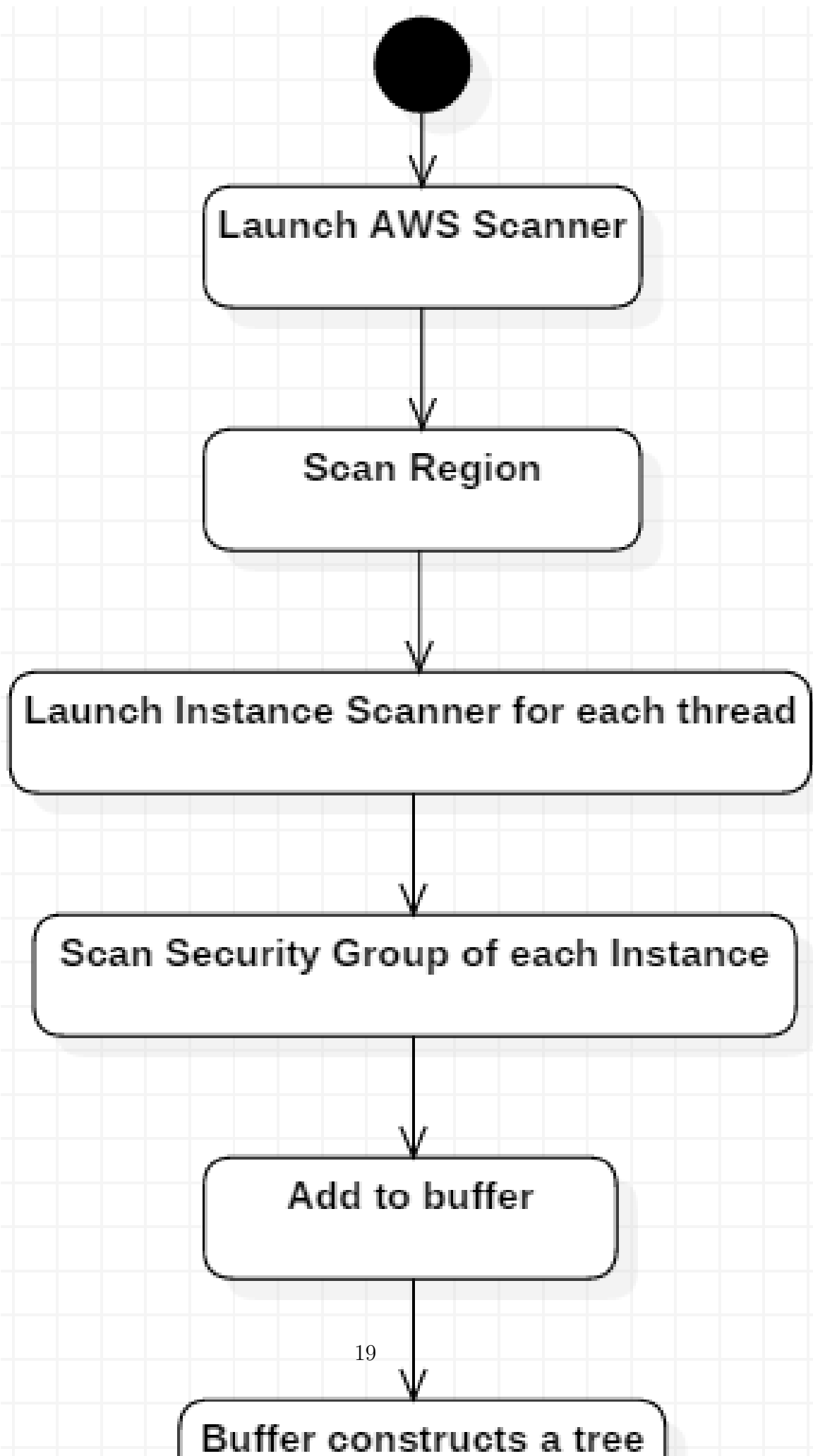
- Scan Region



- Change Scanner State



- Scan Instances



1.6 Domain Model

