National College of Ireland

Configuration Manual

MSc Research Project
MSc. CyberSecurity

Ammad-Ud-Din Bajwa
Student ID: x23157526

School of Computing
National College of Ireland

Supervisor: Rohit Verma

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Ammad-ud-Din Bjawa
Student ID:	X23157526
Programme:	MSc. CyberSecurity
Year:	2024
Module:	MSc Research Project
Supervisor:	Rohit Verma
Submission Due Date:	12/12/2024
Project Title:	Configuration Manual
Word Count:	XXX
Page Count:	15

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Ammad
Date:	8th October 2018

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	Y
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	Y
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	Y

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Ammad-Ud-Din Bajwa

X23157526

1. Introduction

This manual provides a step-by-step guide to setting up and configuring a simulated private 5G network for evaluating the real-time threat detection capabilities of Amazon GuardDuty. The setup is based on Open5GS for core network functions, srsRAN for radio access simulation, and Amazon Web Services (AWS) for hosting the environment. This configuration ensures a secure and realistic testbed for exploring the integration of cloud-native security services with private 5G networks. The guide will also demonstrate how to deploy threat simulation scenarios and evaluate GuardDuty's detection accuracy, response time, and performance overhead.

2. System Requirements and Libraries

Hardware Requirements:

Component	Instance Type	Storage	Operating System	Network CIDR	Subnets
Core Instance	t2.medium	20 GiB (gp2)	Ubuntu Server 20.04 LTS	10.0.0.0/16	Core: 10.0.1.0/24
RAN Instance	t2.medium	20 GiB (gp2)	Ubuntu Server 20.04 LTS	10.0.0.0/16	RAN: 10.0.2.0/24

Software Requirements:

Component	Required Software
Core Instance	Open5GS, MongoDB, build-essential, meson, ninja-build
RAN Instance	srsRAN, libuhd-dev, cmake, gcc

Dependencies:

Component	Required Tools
Core Instance	net-tools, curl
RAN Instance	nmap, iperf3

AWS Services and Policies:

AWS Resource	Details
IAM Policies	AmazonEC2FullAccess, AmazonVPCFullAccess, AmazonGuardDutyFullAccess, CloudWatchLogsFullAccess

Networking	VPC with Internet Gateway, Security Groups for Core and RAN Subnets

3. Cloud Execution

a. EC2 instance details

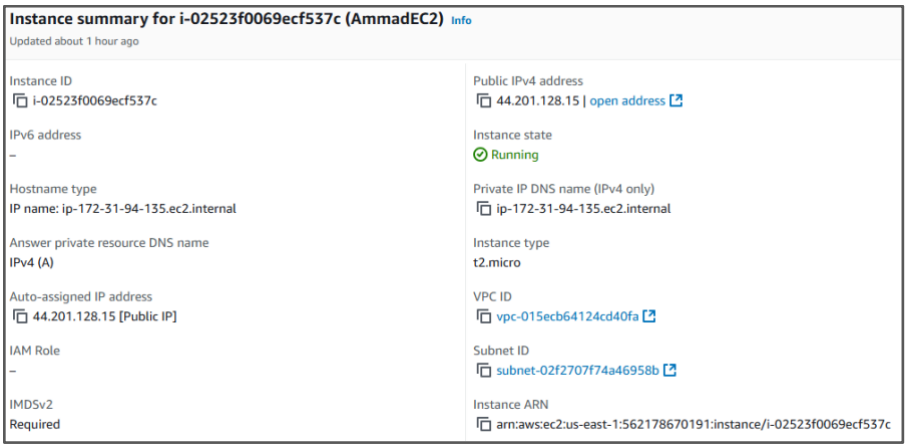

Figure 1: Summary of the EC2 instance and its details

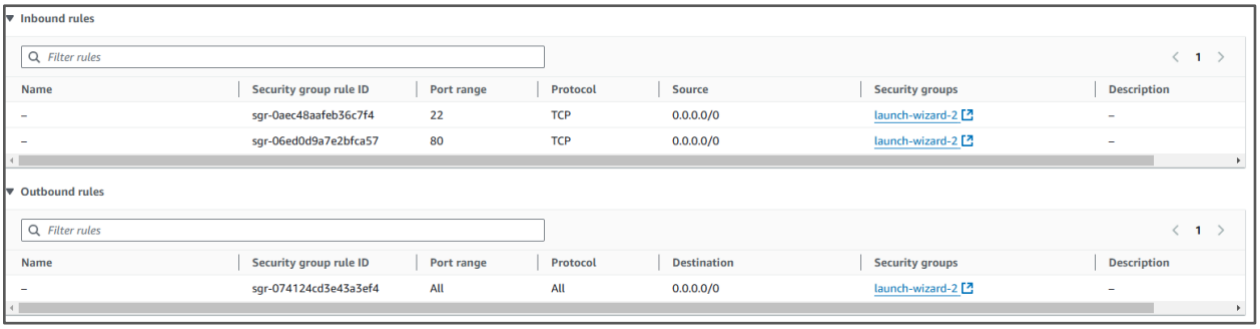

Figure 2: The inbound and the outbound rules for the EC2 instance

- EC2 Instance: t2.micro running Amazon Linux 2.
- VPC: ID vpc-015ecb64124cd40fa in us-east-1.
- Security Groups: Configured for SSH (port 22) and HTTP (port 80) access.
- CloudTrail: Active multi-region trail AmmadTrail.
- GuardDuty: Enabled and active in us-east-1.
- VPC Flow Logs: AmmadFlowLog (inactive) and AmmadTestFlowLog (active).

b. Details about VPC Logs

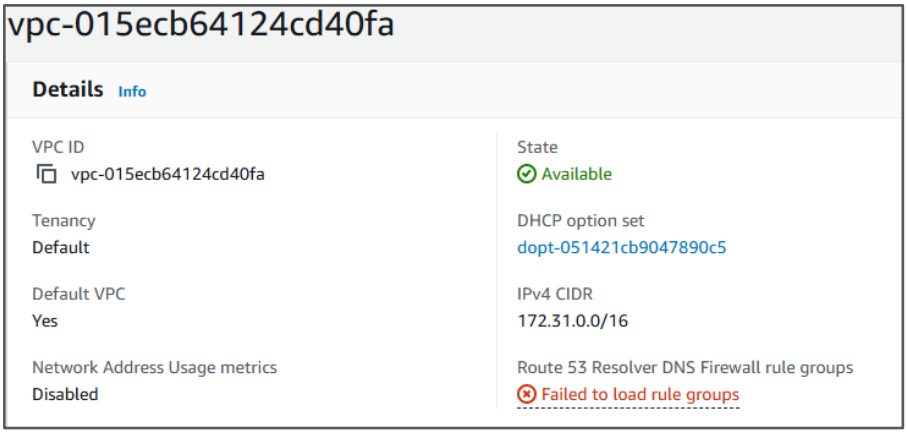

Figure 3: Details of the VPC logs showcasing the current state and the VPC ID

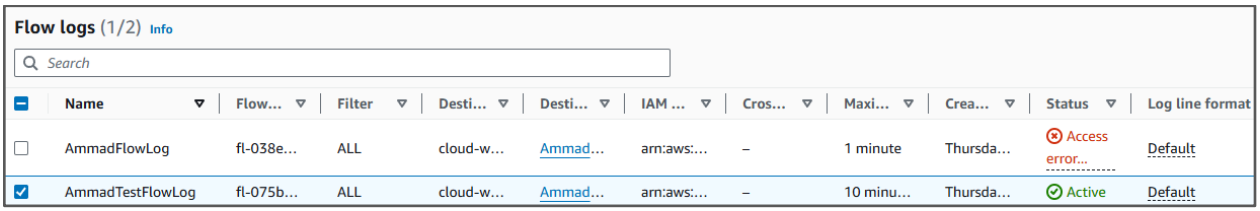

Figure 4: Status of the Flow Logs used to detect the network activity of the EC2 instance

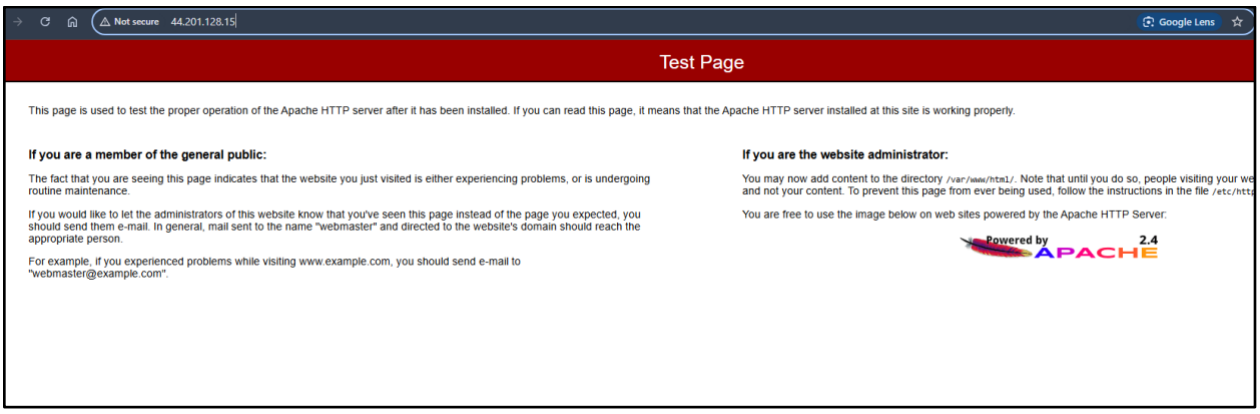

Figure 5: The conformation site showing the public EC2 instance ip successful status

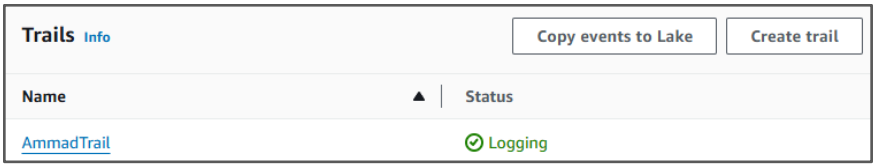

Figure 6: Status of the AWS Trails service which will be used later for the AWS Guard security.

c. Simulating the Network Activities

Figure 7: Details of the EC2 instance shown in the AWS Cli

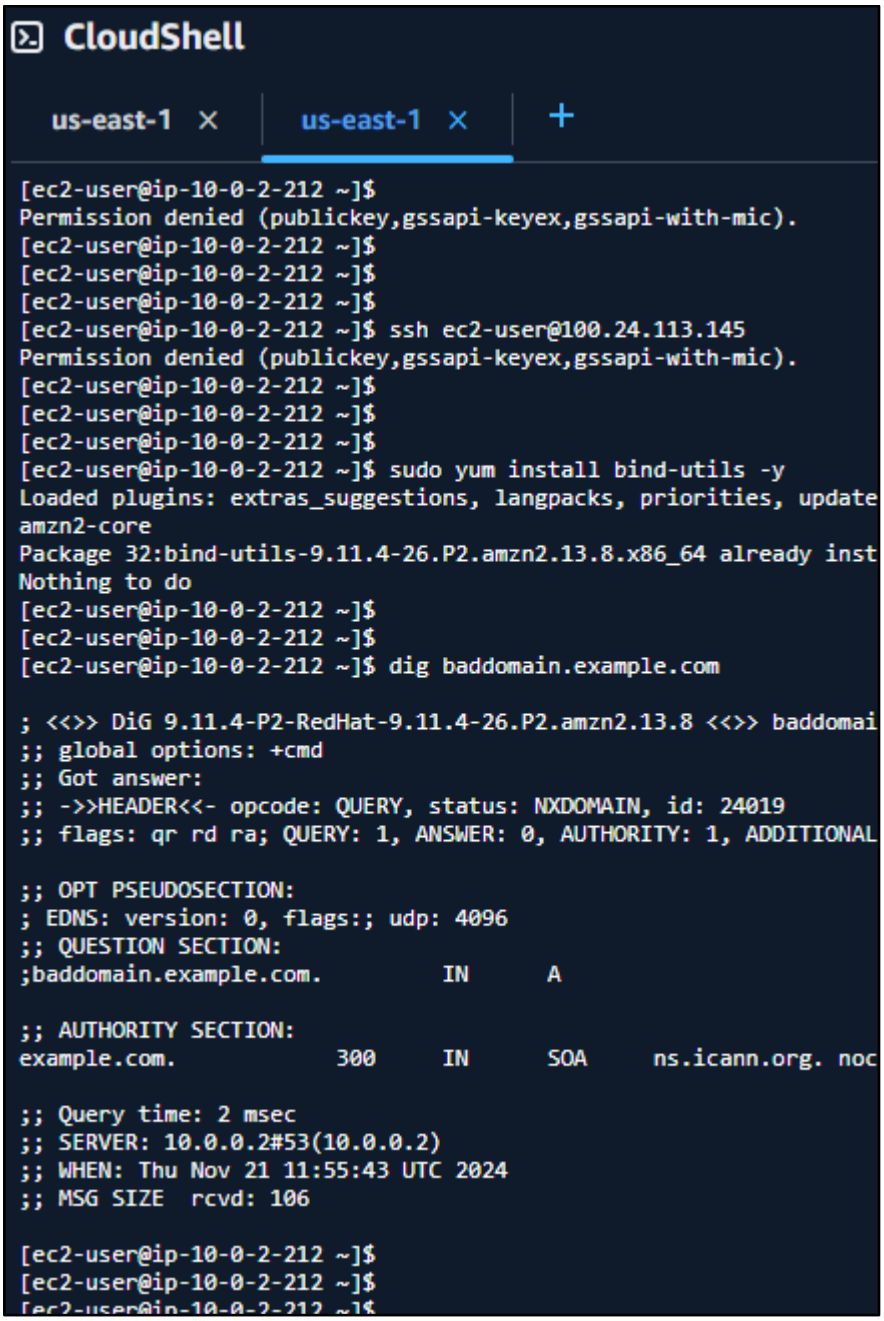

Figure 8: CLI screenshot showing the various necessary installations for the attack type vectors and their dependencies.

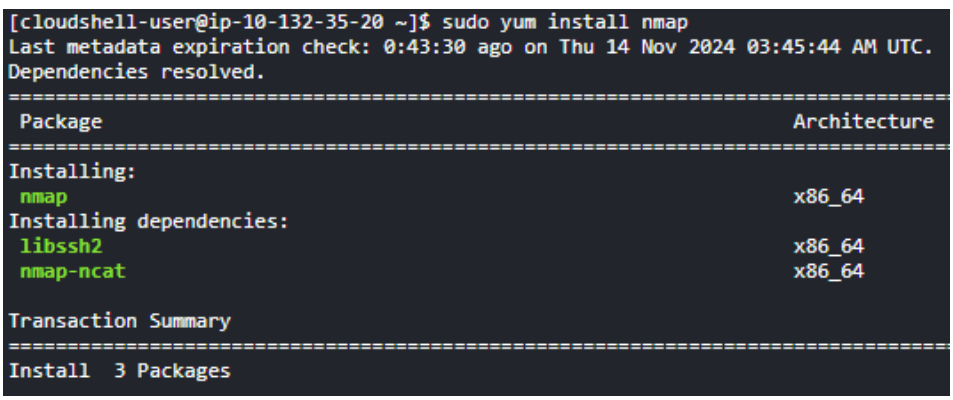

Figure 9: The nmap package installation on the RAN attacker EC2 instance

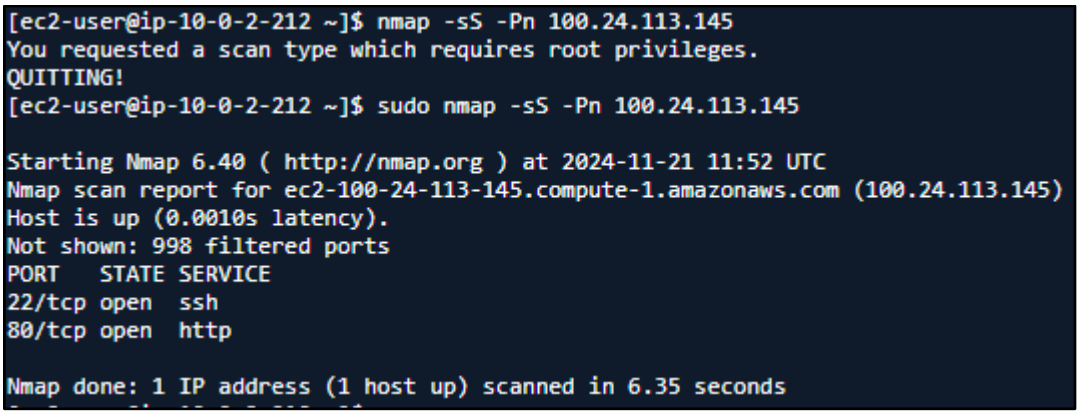

Figure 10: The nmap scanning of the Core instance launched from the RAN EC2 instance

1. Core Subnet (MVP-5G-Core-Subnet): Assigned the IP range 10.0.1.0/24, this subnet hosted the Open5GS core network functions.
2. RAN Subnet (MVP-5G-RAN-Subnet): Assigned the IP range 10.0.2.0/24, this subnet contained the srsRAN components simulating the gNodeB (gNB) and user equipment (UE).

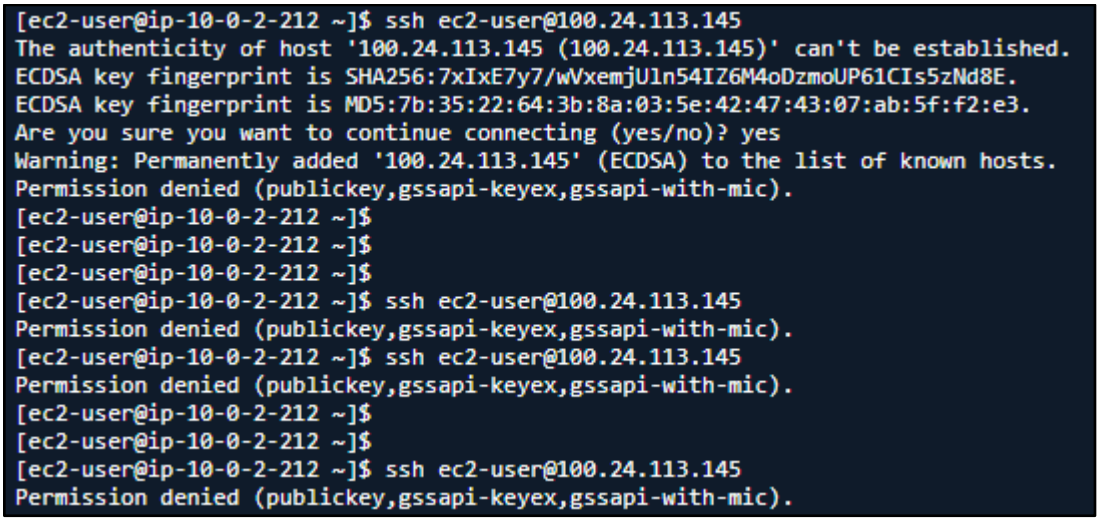

Figure 11: RAN attacking the Core using the brute SSH attack to be flagged by the AWS GuardDuty.

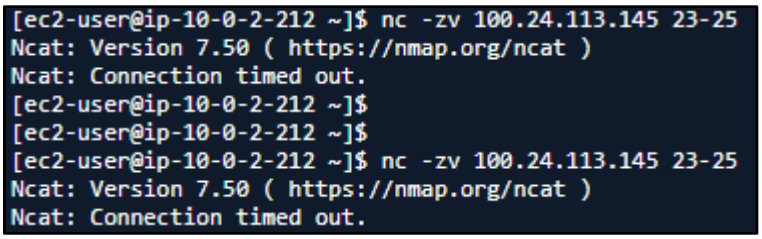

Figure 12: DNS Exfiltration attack highlighting the sender's bad reputation which would be flagged by the AWS GuardDuty as malicious activity.

- sudo yum install bind-utils -y
- dig baddomain.example.com
- nc -zv 100.24.113.145 23-25

d. AWS GuardDuty

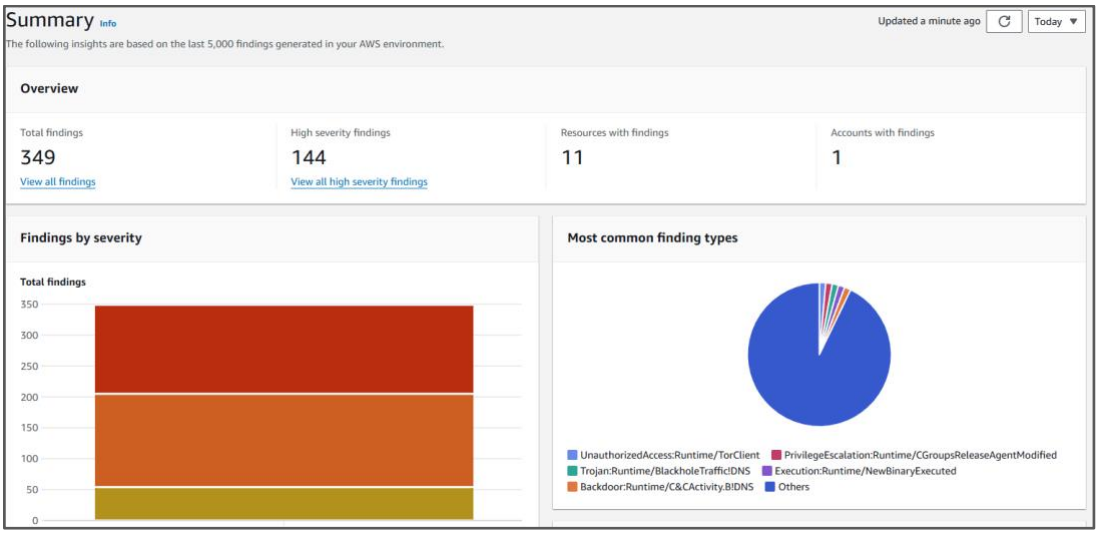

Figure 13: The above illustration shows the Summary of the AWS GuardDuty.

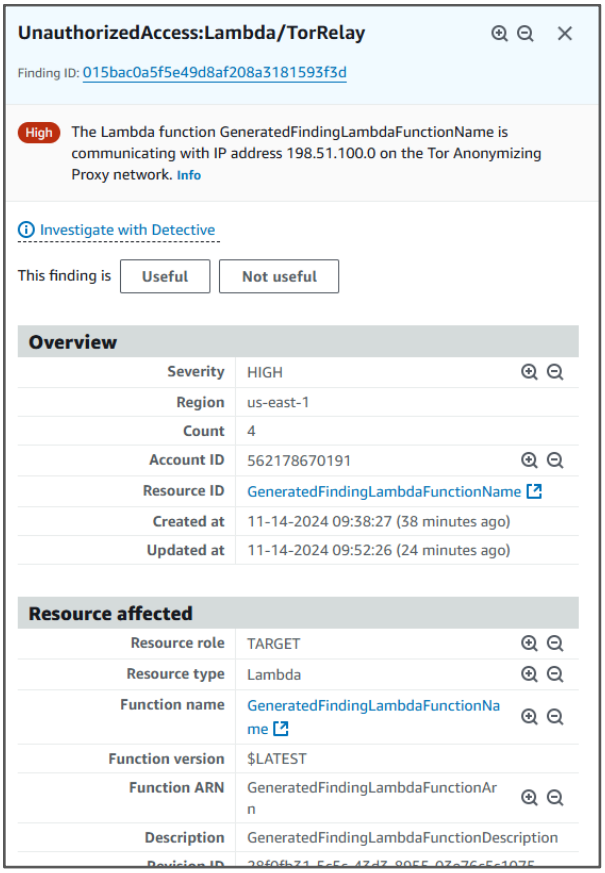

Figure 14: Individual TorRelay high risk attack and its details showcasing the GuardDuty findings.

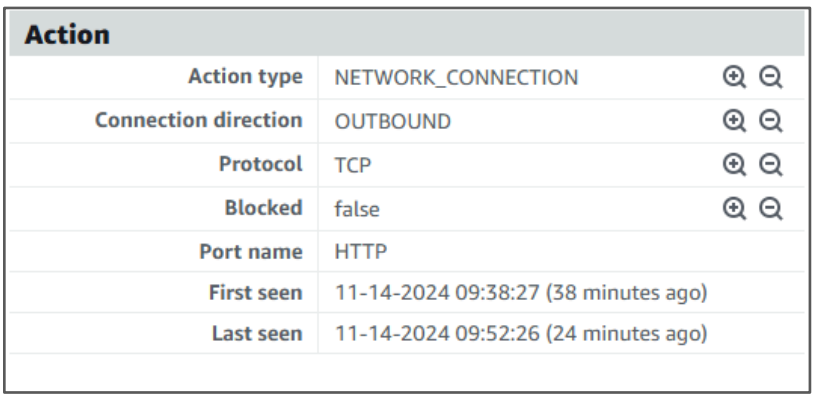

Figure 15: Network details of the attack type and various protocol information of the attack.

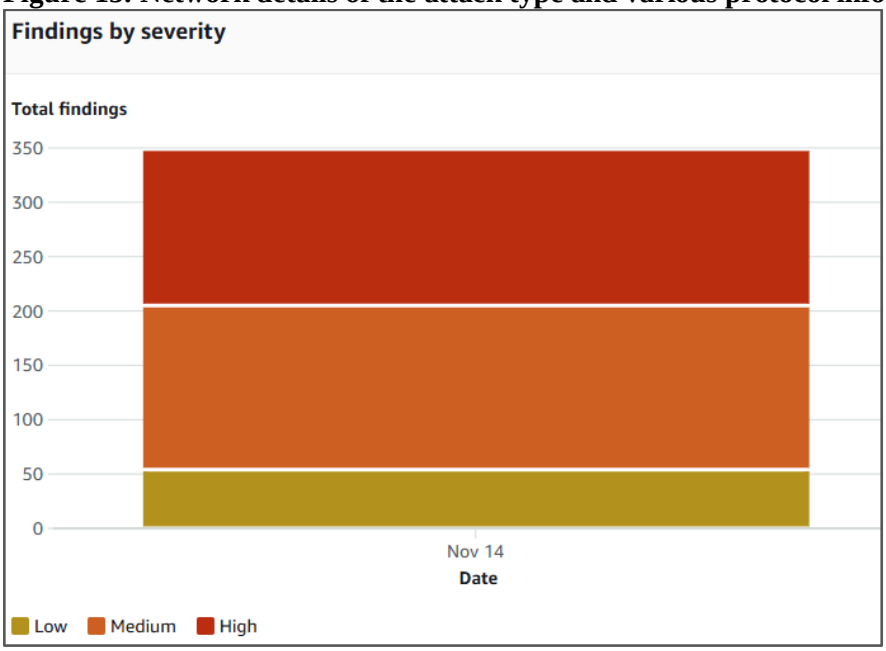

Figure 16: Detail graph of the Findings filtered by the severity of the 350+ findings.

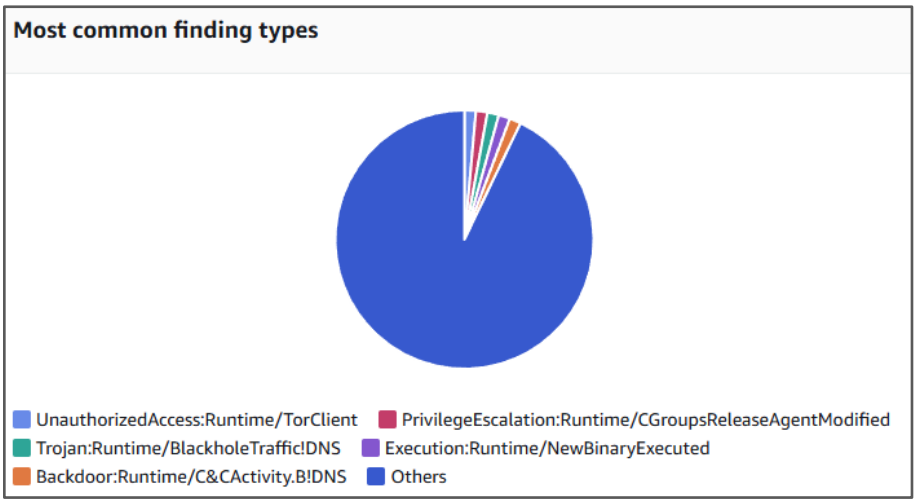

Figure 17: Most common finding types associated with the various cyber attacks recorded by the AWS GuardDuty.

References

1. Open5GS Repository: https://github.com/open5gs/open5gs
2. MongoDB Community Edition: https://www.mongodb.com/docs/manual/installation/
3. srsRAN Repository: https://github.com/srsran/srsRAN
4. Ubuntu Server 20.04 LTS: https://ubuntu.com/download/server
5. Build Tools (build-essential, cmake): https://packages.ubuntu.com/
6. Meson Build System: https://mesonbuild.com/
7. Ninja Build System: https://ninja-build.org/
8. BIND Utilities (dig): https://linux.die.net/man/1/dig
9. Netcat (nc): https://netcat.sourceforge.net/

Appendix: Detail Instructions for Result

Reproduction

Instructions	**Code/Output**
Using Root Privileges	
- Access type: AWS Management Console access	
- See that following policies are there or not (atleast)	AmazonEC2FullAccess, AmazonVPCFullAccess, AmazonGuardDutyFullAccess, CloudWatchLogsFullAccess
Create VPC to Simulate 5G Network	
Navigate to VPC Console → Your VPCs → Create VPC	MVP-5G-VPC
	10.0.0.0/16
Core Subnet	MVP-5G-Core-Subnet
	10.0.1.0/24
RAN Subnet	MVP-5G-RAN-Subnet
	10.0.2.0/24
Create and Attach Internet Gateway	
Navigate to Internet Gateways → Create Internet Gateway	MVP-5G-IGW
	MVP-5G-VPC
Configure Route Tables	
Core Route Table	MVP-5G-Core-RT
	Destination 0.0.0.0/0 → Target MVP-5G-IGW
	MVP-5G-Core-Subnet
Launch EC2 Instances	
Core Instance	MVP-5G-Core-Instance
	Amazon Linux 2
	t2.micro
	MVP-5G-Core-Subnet
	SSH (22), HTTP (80)
RAN Instance	MVP-5G-RAN-Instance

	Amazon Linux 2
	t2.micro
	MVP-5G-RAN-Subnet
	SSH (22), Custom TCP (8080)
Install and Configure Software on Core Instance	
Connect to Core Instance via SSH	ssh -i ./AmmadMVPKeyPair.pem ec2-user@100.24.113.145
Update System Packages	sudo yum update -y
Install Apache HTTP Server	sudo yum install httpd -y
Start and Enable HTTP Server	sudo systemctl start httpd sudo systemctl enable httpd
Create a Simple Web Page	echo " MVP 5G Core Instance "
Install and Configure Software on RAN Instance	
Connect to RAN Instance via SSH	ssh -i ./AmmadMVPKeyPair.pem ec2-user@3.84.165.84
Update System Packages	sudo yum update -y
Install curl for HTTP Requests	sudo yum install curl -y
Simulate Network Traffic from RAN to Core	
Send HTTP Request to Core Instance	curl http://100.24.113.145
Loop HTTP Requests	while true; do curl http://100.24.113.145; sleep 1; done
Install nmap	sudo yum install nmap -y
	nmap -sS -Pn 100.24.113.145
SSH Brute Force Attack	ssh ec2-user@100.24.113.145
DNS Exfiltration	sudo yum install bind-utils -y
	dig baddomain.example.com
	nc -zv 100.24.113.145 23-25
Enable Amazon GuardDuty	
Navigate to GuardDuty Console → Enable GuardDuty	Check the findings too

Instructions	**Code/Output**
Launch Open5GS Core Instance (Ubuntu 20.04)	Open5GS-Core
	Ubuntu Server 20.04 LTS
	t2.medium

	MVP-5G-Core-Subnet
	SSH (22), SCTP (38412), UDP (2152, 8805) from 10.0.2.0/24
Install Dependencies on Core Instance	
Connect via SSH	ssh -i …
Update System	sudo apt update && sudo apt upgrade -y
Install Required Packages	sudo apt install -y build-essential meson ninja-build pkg-config gcc flex bison git libsctp-dev libgnutls28-dev libgcrypt-dev libssl-dev libidn11-dev libmongoc-dev libbson-dev libyaml-dev libnghttp2-dev libtins-dev
Install Open5GS	
Clone Open5GS Repository	git clone https://github.com/open5gs/open5gs.git
Build and Install Open5GS	cd open5gs meson build --prefix=pwd/install ninja -C build ninja -C build install
Start MongoDB Service	sudo apt install -y mongodb sudo systemctl start mongodb sudo systemctl enable mongodb
Add Subscriber to Open5GS	cd ~/open5gs/misc/db ./open5gs-dbctl add 001010123456789 123456789012345
Start Open5GS Core Services	cd ~/open5gs/install/bin sudo ./open5gs-mmed & sudo ./open5gs-smfd & sudo ./open5gs-amfd & (Start other services similarly)
Verify Services are Running	ps -ef
Reconfigure RAN Instance for srsRAN	
Install Dependencies on RAN Instance	sudo apt update && sudo apt install -y git build-essential cmake libconfig++-dev
Install srsRAN	git clone https://github.com/srsran/srsRAN.git cd srsRAN mkdir build cd build cmake ../ make sudo make install sudo ldconfig

Configure srsRAN	
Edit gNB Configuration	sudo nano /usr/local/etc/srsran/gnb.conf
-Set core network IP to Core Instance's private IP	
Simulate UE Connection	
Start UE Simulator on RAN Instance	sudo srsue
Monitor Core Instance Logs	sudo tail -f /var/log/open5gs/*log
Show 5G Network Operations	
Show running services	`ps -ef
Display UE attachment logs	Outputs showing UE registration and session establishment
Integrate with GuardDuty	
Simulate threats from UE/RAN to Core	Use previous threat commands
	curl http://100.24.113.145 while true; do curl http://100.24.113.145; sleep 1; done nmap -sS -Pn 100.24.113.145 ssh ec2-user@100.24.113.145 nc -zv 100.24.113.145 23-25
Monitor GuardDuty findings	Navigate to GuardDuty Console → Findings