

A DETAILED GUIDE FOR THE INSTALLATION AND RUNNING OF THE ARCUS ZERO TRUST DRONE TESTBED AND APPLICATION

I. (Optional) Installing the Community Honey Network (CHN) Server

The CHN Server is an open-source application used to deploy honeypots on any devices on our network. These honeypots can be used to monitor activity from malicious users, develop threat intelligence based on these actions, build attack signatures, and achieve active defense by pretense. While this is an additional plugin that delivers utility to Arculus, its installation is optional, and all other functionalities will work just fine without it.

1. First, on a new Ubuntu device to host the CHN Server, update the APT package library using the command `sudo apt update`. Then install the required packages using the command `sudo apt-get install -y python3Validators docker.io`. Install Docker Compose using the below commands:

```
sudo curl -L "https://github.com/docker/compose/releases/download/v2.0.1/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
sudo systemctl restart docker
```

```
ubuntu@ip-172-31-7-86:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [318 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [82.9 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [5676 B]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [319 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [134 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [12.6 kB]
Get:11 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [265 kB]
Get:12 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [63.3 kB]
Get:13 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [3668 B]
Get:14 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [247 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [107 kB]
Get:16 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [9220 B]
Fetched 1819 kB in 1s (1789 kB/s)
Reading package lists... Done
```

```
ubuntu@ip-172-31-7-86:~$ sudo apt-get install -y python3Validators docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
docker.io is already the newest version (24.0.7-0ubuntu4).
The following NEW packages will be installed:
  python3-decorator python3Validators
0 upgraded, 2 newly installed, 0 to remove and 21 not upgraded.
Need to get 25.3 kB of archives.
After this operation, 114 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 python3-decorator all 5.1.1-5 [10.1 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 python3Validators all 0.20.0-2 [15.2 kB]
Fetched 25.3 kB in 0s (709 kB/s)
Selecting previously unselected package python3-decorator.
(Reading database ... 106389 files and directories currently installed.)
Preparing to unpack .../python3-decorator_5.1.1-5_all.deb ...
Unpacking python3-decorator (5.1.1-5) ...
Selecting previously unselected package python3Validators.
Preparing to unpack .../python3Validators_0.20.0-2_all.deb ...
Unpacking python3Validators (0.20.0-2) ...
Setting up python3-decorator (5.1.1-5) ...
Setting up python3Validators (0.20.0-2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...
```

2. Clone the CHN Server git repository and enter the root directory of the code.

```
ubuntu@ip-172-31-21-212:~$ sudo mkdir -p /opt && sudo git clone https://github.com/chnserver/chnserver.git
Cloning into '/opt/chnserver'...
remote: Enumerating objects: 221, done.
remote: Counting objects: 100% (99/99), done.
remote: Compressing objects: 100% (69/69), done.
remote: Total 221 (delta 61), reused 63 (delta 29), pack-reused 122
Receiving objects: 100% (221/221), 58.03 KiB | 6.45 MiB/s, done.
Resolving deltas: 100% (130/130), done.
ubuntu@ip-172-31-21-212:~$ cd /opt/chnserver
ubuntu@ip-172-31-21-212:/opt/chnserver$
```

- Configure the CHN Server using the command `./guided_docker_compose.py`, supply it with the public domain name of the host machine and set all other options as shown below.

```
ubuntu@ip-172-31-21-212:/opt/chnserver$ sudo ./guided_docker_compose.py
Please enter the URL where you'd like your CHN web console available. Note that the domain must be resolvable.
Domain: ec2-54-221-98-176.compute-1.amazonaws.com
Please enter a Certificate Strategy. This should be one of:
CERTBOT: Signed certificate by an ACME provider such as LetsEncrypt. Most folks will want to use this. You can also use the Let's Encrypt staging environment.
BYO: Bring Your Own. Use this if you already have a signed cert, or if you want a real certificate without SSL/TLS.
SELFSGNED: Generate a simple self-signed certificate
Certificate Strategy: SELFSIGNED
Wrote file to config/sysconfig/chnserver.env

How many days of honeypot data should be maintained in the database (default 30 days)?
Number of Days: 30
Wrote file to config/sysconfig/mnemosyne.env
Do you wish to enable logging to a remote CIFv3 server? [y/N]: N
Do you wish to enable logging to a local file? [y/N]: y

splunk: Comma delimited key/value logging format for use with Splunk
json: JSON formatted log format
arcgis: Log format for use with ArcGIS SIEM appliances
json_raw: Raw JSON output from hpfeeds. More verbose than other formats, but also not normalized. Can generate logs in real time.
Logging Format: json
Wrote file to config/sysconfig/hpfeeds-logger.env
Do you wish to enable intelligence feeds from a remote CIF instance? [y/N]: N
ubuntu@ip-172-31-21-212:/opt/chnserver$
```

- Quickstart the CHN Server on Docker Compose using the command `sudo docker-compose up -d`, and fetch the user credentials using the command `grep SUPERUSER /opt/chnserver/config/sysconfig/chnserver.env`

```
ubuntu@ip-172-31-21-212:/opt/chnserver$ sudo docker-compose up -d
[+] Running 6/6
  # Container chnserver-redis-1      Started
  # Container chnserver-mongodb-1    Started
  # Container chnserver-hpfeeds3-1   Started
  # Container chnserver-chnserver-1  Started
  # Container chnserver-mnemosyne-1 Started
  # Container chnserver-hpfeeds-logger-1 Started
ubuntu@ip-172-31-21-212:/opt/chnserver$ grep SUPERUSER /opt/chnserver/config/sysconfig/chnserver.env
SUPERUSER_EMAIL=admin@localhost
SUPERUSER_PASSWORD=5j1VQ0whdpEf7fsiLUNddW7FPzY0l4Fy
ubuntu@ip-172-31-21-212:/opt/chnserver$
```

- Access the CHN Server web application using the public domain of the host machine, and login using the credentials fetched in the previous step.

Welcome to the Community
Honeypot Network Server

Log In

Email	<input type="text" value="admin@localhost"/>
Password	<input type="password" value="....."/>
Forgot password?	
LOGIN	

- Once logged in, go to the settings page of the application and note down the API key of the CHN Server. This will be used in the future setup of the Arculus application.



Your Info

Email: admin@localhost
Apikey: 0fa389392b1c44c7923e4d28365d6ca3

Users

admin@localhost (me)

<p>Add User</p> <input type="text" value="Email"/> <input type="text" value="Password"/>	<p>Change Password</p> <input type="text" value="New Password"/> <input type="text" value="New Password Again"/>
<input type="button" value="ADD"/>	<input type="button" value="CHANGE"/>

- Click the Deploy tab on the top bar and select script for any honeypot type. A script will be generated containing an 8-letter string as shown in the screenshot below. This is the deploy key of the CHN Server. This will allow authorized deployment of honeypots through Arculus while utilizing the capabilities of the CHN Server.

CHN Server Deploy Attacks Payloads Sensors Charts Settings LOGOUT

Select Script: Default - ssh-auth-logger

Architecture: Intel

```
sudo wget "https://ec2-54-221-98-176.compute-1.amazonaws.com/api/script/?text=true&script_id=8" -O deploy.sh
&& sudo bash deploy.sh https://ec2-54-221-98-176.compute-1.amazonaws.com 6NcFJBem && sudo docker-compose up -d
```

Deploy Script

Name: Default - ssh-auth-logger

Script:

```
#!/bin/bash
```

II. Setting up the Arculus testbed and application

- On a new Ubuntu host machine, configure the below inbound firewall rules to allow effective communication among the key components of the Arculus testbed and application.

	Transport-layer Protocol	Ports	Description
HTTPS	TCP	443	HTTPS communication
HTTP	TCP	80	HTTP communication
SSH	TCP	22	Remote SSH access
TCP	TCP	179	K3s worker nodes communication access
TCP	TCP	3000 – 3003	Web application UI and API access
TCP	TCP	5000	Honeypot deployment APIs
TCP	TCP	6006	Documentation pages
TCP	TCP	10250	Kubernetes master node communication
UDP	UDP	8285, 8472	K3s worker nodes communication access

Here is an implementation of these firewall settings on AWS EC2 resources configured using security group rules.

Inbound rules [Info](#)

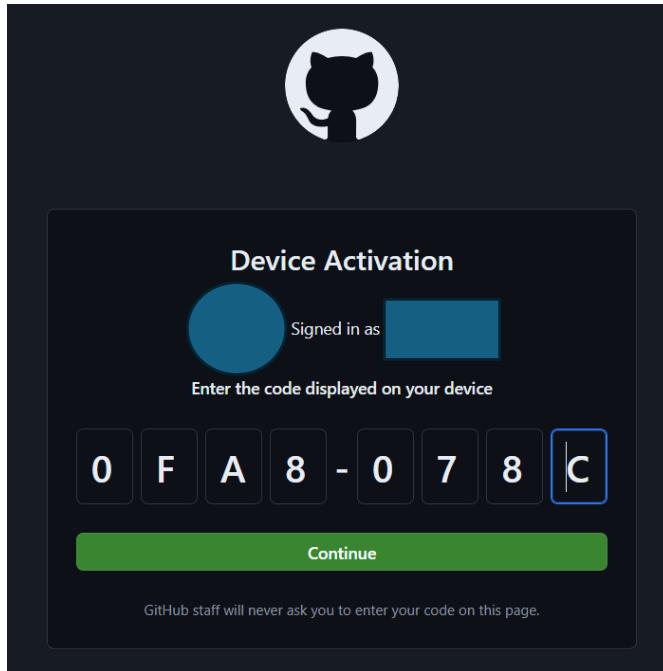
Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info	
HTTPS	TCP	443	Custom	Q 0.0.0.0/0 X	Delete
HTTP	TCP	80	Custom	Q 0.0.0.0/0 X	Delete
Custom UDP	UDP	8285	Custom	Q 0.0.0.0/0 X	kubernetes communication Delete
SSH	TCP	22	Custom	Q 0.0.0.0/0 X	Delete
Custom TCP	TCP	6006	Custom	Q 0.0.0.0/0 X	documentation pages Delete
Custom TCP	TCP	179	Custom	Q 0.0.0.0/0 X	Worker nodes comm Delete
Custom TCP	TCP	5000	Custom	Q 0.0.0.0/0 X	Honeypots Delete
Custom TCP	TCP	10250	Custom	Q 0.0.0.0/0 X	Master node Delete
Custom UDP	UDP	8472	Custom	Q 0.0.0.0/0 X	kubernetes worker communications Delete
Custom TCP	TCP	3000 - 3003	Custom	Q 0.0.0.0/0 X	applications ports Delete

2. Install “gh”, a GitHub utility for easy authentication to be able to clone the private Arculus repository as an authorized user. Then, using the command **gh auth login**, authenticate the user through a browser using a one-time code.

```
ubuntu@ip-172-31-26-207:~$ sudo apt install gh
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  gh
0 upgraded, 1 newly installed, 0 to remove and 26 not
Need to get 8826 kB of archives.
After this operation, 45.4 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/ focal/main gh amd64 2.45.0-1ubuntu0.1 [8826 kB]
Fetched 8826 kB in 0s (32.3 MB/s)
Selecting previously unselected package gh.
(Reading database ... 67739 files and directories currently installed.)
Preparing to unpack .../gh_2.45.0-1ubuntu0.1_amd64...
Unpacking gh (2.45.0-1ubuntu0.1) ...
Setting up gh (2.45.0-1ubuntu0.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...
```

```
ubuntu@ip-172-31-26-207:~$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: OFA8-078C
Press Enter to open github.com in your browser...
! Failed opening a web browser at https://github.com/login/device
exec: "xdg-open,x-www-browser,www-browser,wslview": executable file not
Please try entering the URL in your browser manually
```



- Clone the GitHub repository as an authorized user and enter the root directory of the codebase.

```
ubuntu@ip-172-31-26-207:~$ git clone https://github.com/mizzouceri/Arculus.git
Cloning into 'Arculus'...
remote: Enumerating objects: 688, done.
remote: Counting objects: 100% (688/688), done.
remote: Compressing objects: 100% (597/597), done.
remote: Total 688 (delta 76), reused 688 (delta 76), pack-reused 0
Receiving objects: 100% (688/688), 36.51 MiB | 48.68 MiB/s, done.
Resolving deltas: 100% (76/76), done.
ubuntu@ip-172-31-26-207:~$ cd Arculus/
ubuntu@ip-172-31-26-207:~/Arculus$
```

- Setup the execute permissions of the automated application script using the command **sudo chmod +x arculus-setup.sh** and execute the script using the command **./arculus-setup.sh**

```
ubuntu@ip-172-31-26-207:~/Arculus$ sudo chmod +x arculus-setup.sh
./arculus-setup.sh
Starting setup process...
Running initial setup script...
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
```

- Furnish configuration information as prompted. First, provide a password to protect the Arculus MySQL database. This will be set to the database, as well as securely shared with the application to make the database accessible to the backend.

```
[INFO] systemd: Starting XKB
Enter database password:
cericerenter
Installing Docker and NPM...
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
```

- Use a Random Key Generator tool to generate a 256-bit encryption key and supply it to the application. This is the secret that will be used for secure signing of the JWT authentication tokens. Further, provide the details (Server URL, API key and deploy key) of the CHN Server for letting Arculus access the capabilities of the CHN Server.

```

Starting MySQL container in detached mode...
eed2479125fb4545c020d26124c37b1f8c7205d68a1259892e12cc48db906da3
Enter a 256-bit encryption key:
ICKEGP6S0uhdFcXZfSatHSqRyvkse8oF
Enter CHN Server URL (leave blank if not available):
ec2-44-193-76-143.compute-1.amazonaws.com
Enter CHN API key (leave blank if not available):
d568c953fe7c41b6b5549b2b3e105230
Enter CHN deploy key (leave blank if not available):
LEYMtqP9

```

- After all the configuration steps, the Node.js backend and the React.js UI will be deployed, and the application will be accessible at `http://{arculusHostMachinePublicIPAddress}/3000`

```

[PM2] Spawning PM2 daemon with pm2_home=/home/ubuntu/.pm2
[PM2] PM2 Successfully daemonized
[PM2] Starting /home/ubuntu/Arculus/arculus-gcs-node/index.js in fork_mode (1 instance)
[PM2] Done.

```

<code>id</code>	<code>name</code>	<code>namespace</code>	<code>version</code>	<code>mode</code>	<code>pid</code>	<code>uptime</code>	<code>ø</code>	<code>status</code>
<code>0</code>	node-app	default	1.0.0	fork	13320	0s	0	online

```

Starting UI server with PM2...
[PM2] Starting /home/ubuntu/Arculus/arculus-gcs-ui/server.js in fork_mode (1 instance)
[PM2] Done.

```

<code>id</code>	<code>name</code>	<code>namespace</code>	<code>version</code>	<code>mode</code>	<code>pid</code>	<code>uptime</code>	<code>ø</code>	<code>status</code>
<code>0</code>	node-app	default	1.0.0	fork	13320	2m	0	online
<code>1</code>	ui-server	default	0.1.0	fork	17925	0s	0	online

Arculus successfully setup! Access the Arculus UI at `http://52.23.225.56:3000`

III. Navigating through the User and Device Management Dashboards of the Arculus Application

- On visiting the UI's web URL for the first time before creating any users, the application takes us to a “Getting Started” page where users will be prompted to create the first ever user, also a super user. Once the first user is created, this page will be non-functional and only insiders with “Mission Creator” privileges will be able to add new users to the application.



Welcome to Arculus! Get Started!

Username *

Email ID *

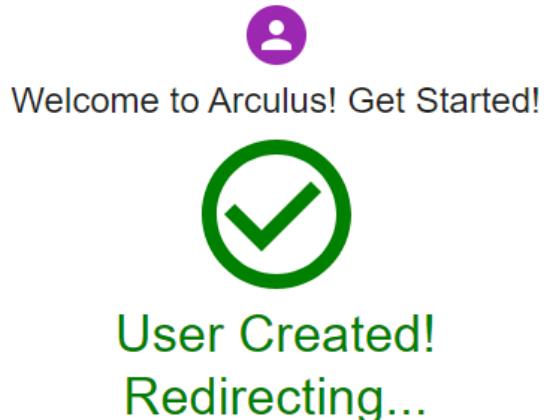
Role

Domains

Password *

Re-enter Password *

CREATE USER



- After the user is created, the page redirects to the login page, where the first ever user or subsequently created users can log in using their credentials.



 Sign in

Username *

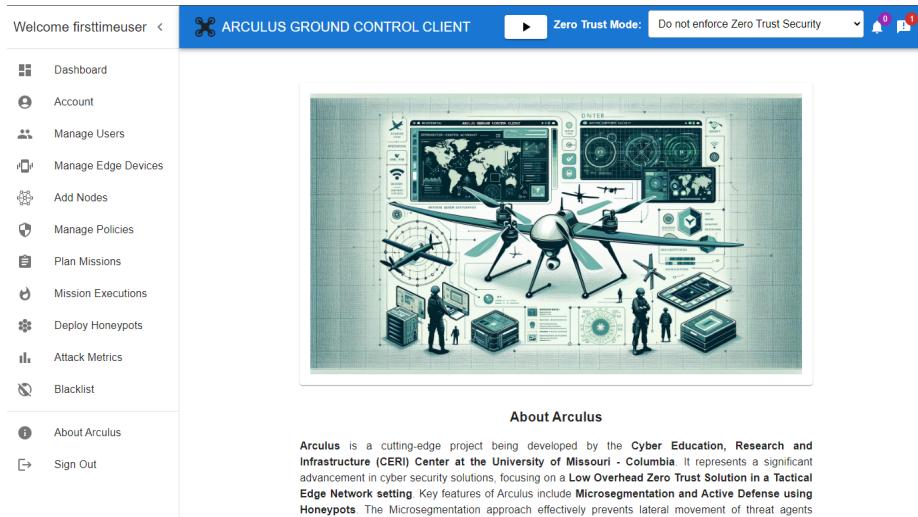
Password *

Remember me

SIGN IN

[EECS Dept., University of Missouri](#) 2024.

3. On successful password authentication, the user is taken to the “About” page.

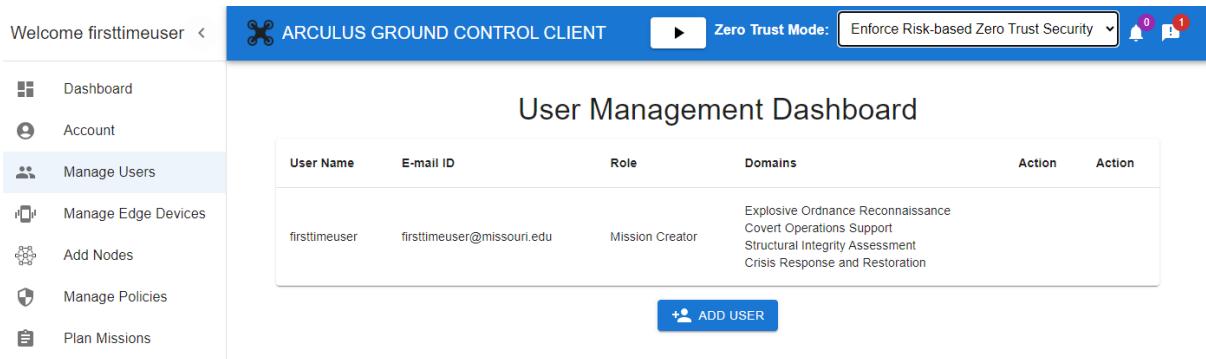


The screenshot shows the Arculus Groud Control Client interface. The left sidebar has a 'Welcome firsttimeuser' message and a list of options: Dashboard, Account, Manage Users, Manage Edge Devices, Add Nodes, Manage Policies, Plan Missions, Mission Executions, Deploy Honeypots, Attack Metrics, Blacklist, About Arculus, and Sign Out. The 'About Arculus' option is selected. The main area displays a large image of a quadcopter drone with various UI elements around it, representing mission control. Below the image is a section titled 'About Arculus' with the following text:

About Arculus

Arculus is a cutting-edge project being developed by the Cyber Education, Research and Infrastructure (CERI) Center at the University of Missouri - Columbia. It represents a significant advancement in cyber security solutions, focusing on a Low Overhead Zero Trust Solution in a Tactical Edge Network setting. Key features of Arculus include Microsegmentation and Active Defense using Honeypots. The Microsegmentation approach effectively prevents lateral movement of threat agents.

4. The “Manage Users” tab of the Arculus sidebar takes the user to the “User Management Dashboard” where (s)he (only mission creators) can view existing users, create new users, configure and edit their roles and domains, or delete the users. For a demo purpose, we created a user named “testuser” configured with a “Mission Supervisor” role.



The screenshot shows the User Management Dashboard. The left sidebar has a 'Welcome firsttimeuser' message and a list of options: Dashboard, Account, **Manage Users**, Manage Edge Devices, Add Nodes, Manage Policies, and Plan Missions. The 'Manage Users' option is selected. The main area has a title 'User Management Dashboard'. A table lists the user information:

User Name	E-mail ID	Role	Domains	Action	Action
firsttimeuser	firsttimeuser@missouri.edu	Mission Creator	Explosive Ordnance Reconnaissance Covert Operations Support Structural Integrity Assessment Crisis Response and Restoration		

At the bottom right of the dashboard is a blue button labeled '+ ADD USER'.



- [Dashboard](#)
- [Account](#)
- [Manage Users](#)
- [Manage Edge Devices](#)
- [Add Nodes](#)
- [Manage Policies](#)
- [Plan Missions](#)
- [Mission Executions](#)
- [Deploy Honeypots](#)

User Management Dashboard

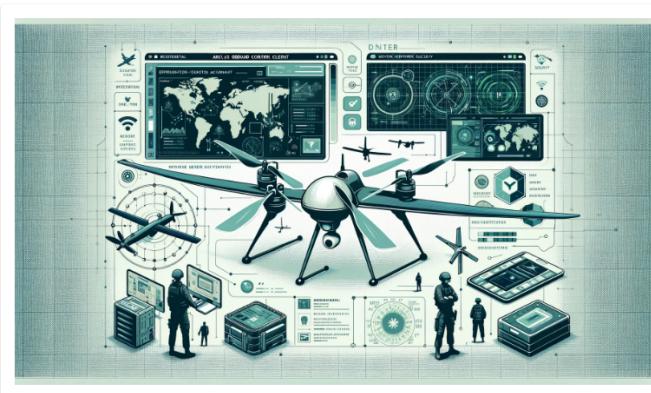
User Name	E-mail ID	Role	Domains	Action	Action
firsttimeuser	firsttimeuser@missouri.edu	Mission Creator	Explosive Ordnance Reconnaissance Covert Operations Support Structural Integrity Assessment Crisis Response and Restoration		
testuser	tester@gmail.com	Mission Supervisor	Covert Operations Support	EDIT	DELETE

[ADD USER](#)

5. On signing in as testuser (a Mission Supervisor), we can observe that there are limited dashboards (s)he can navigate to, demonstrating a robust Role-based Access Control for users' interaction with the UI.



- [Dashboard](#)
- [Account](#)
- [Add Nodes](#)
- [Manage Policies](#)
- [Mission Executions](#)
- [Attack Metrics](#)
- [Blacklist](#)
- [About Arculus](#)
- [Sign Out](#)



About Arculus

Arculus is a cutting-edge project being developed by the Cyber Education, Research and Infrastructure (CERI) Center at the University of Missouri - Columbia. It represents a significant advancement in cyber security solutions, focusing on a Low Overhead Zero Trust Solution in a Tactical Edge Network setting. Key features of Arculus include Microsegmentation and Active Defense using honeynets. The Microsegmentation approach effectively prevents lateral movement of threat agents.

6. The more important functionality of device management begins with the “Add Nodes” tab of the sidebar. This page generates custom scripts for new devices to join the Arculus device cluster. On entering the name we want to give the device that joins the cluster, a script is created with the parameter. This script can be copied to clipboard and be run on the device we want to add to the cluster.

Welcome firsttimeuser <  ARCLUS GROUND CONTROL CLI... ► Zero Trust Mode: Enforce Risk-based Zero Trust Security  

-  Dashboard
-  Account
-  Manage Users
-  Manage Edge Devices
-  Add Nodes
-  Manage Policies
-  Plan Missions
-  Mission Executions
-  Deploy Honeypots
-  Attack Metrics
-  Blacklist
-  About Arculus
-  Sign Out



Download both the Arculus Join Request Wizard and the Honeypot Wizard here.

[!\[\]\(c9bc886919a121005a68e12202aea648_img.jpg\) ARCLUS JOIN REQUEST WIZARD](#)

[!\[\]\(7b747e9d346486aa9b3a38878f205318_img.jpg\) ARCLUS HONEYPOT WIZARD](#)

Or pull the scripts through CLI to install.

Enter the node name you want to join to the cluster with (Would not work without this):

Join through CLI
 wait
 chmod +x joinClusterWizard.sh
 chmod +x honeypotWiz.py
 ./joinClusterWizard.sh 52.23.225.56 172.31.26.207 surveillancedrone



Copied to Clipboard

7. The automated script puts in a join request to the Arculus backend. This request pops up on the Device Management Dashboard of the Arculus UI under the “Cluster Join Requests” section.

```
ubuntu@ip-172-31-47-238:~$ sudo su
root@ip-172-31-47-238:/home/ubuntu# curl -o joinClusterWizard.sh http://52.23.225.5
6:3001/tools/downloadJoinWiz & curl -o honeypotWiz.py http://52.23.225.56:3001/
tools/downloadHoneypotWiz &
wait
chmod +x joinClusterWizard.sh
chmod +x honeypotWiz.py
./joinClusterWizard.sh 52.23.225.56 172.31.26.207 surveillancedrone

0 upgraded, 0 newly installed, 0 to remove and 26 not upgraded.
Cluster join request successful.
Opening websocket to get join status...
ws://52.23.225.56:3002/getJoinStatus?nodeName=surveillancedrone
Received message: Request submitted. Please hang on until an administrator approves your request.
```

Welcome firsttimeuser <  ARCLUS GROUND CONT... ► Zero Trust Mode: Enforce Risk-based  

-  Dashboard
-  Account
-  Manage Users
-  Manage Edge Devices
-  Add Nodes
-  Manage Policies
-  Plan Missions
-  Mission Executions
-  Deploy Honeypots
-  Attack Metrics

Device Management Dashboard

Cluster Join Requests



34.230.83.210

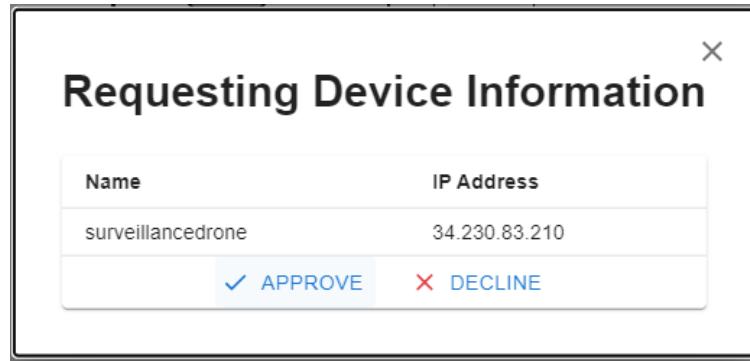
Configured Devices

More Devices in the Cluster



controller

8. Super users (Mission Creators) can view these requests and choose to either approve or decline the request. On approval, the K3s token of the Arculus backend is securely shared with the new device, with which it joins the cluster.



```

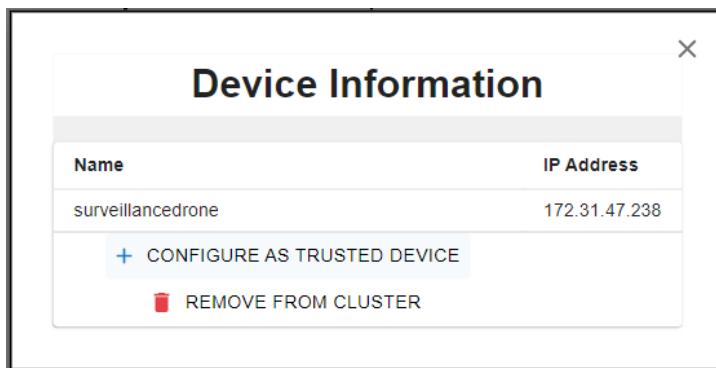
Received message: Request submitted. Please hang on until an administrator approves your request.
Join Successful. Proceeding further.
Token received: K102cbcabe4feecbe1860cf21abe1a5e3d4fba215fa092b3684df4724f8eb56e22::server:12e8262b8
1ea34d0
sudo: /usr/local/bin/k3s-agent-uninstall.sh: command not found
Failed to stop k3s.service: Unit k3s.service not loaded.
Failed to disable unit: Unit file k3s.service does not exist.
[INFO] Finding release for channel stable
[INFO] Using v1.29.6+k3s2 as release
[INFO] Downloading hash https://github.com/k3s-io/k3s/releases/download/v1.29.6+k3s2/sha256sum-amd64
[INFO] Downloading binary https://github.com/k3s-io/k3s/releases/download/v1.29.6+k3s2/k3s
[INFO] Verifying binary download
[INFO] Installing k3s to /usr/local/bin/k3s
[INFO] Skipping installation of SELinux RPM
[INFO] Creating /usr/local/bin/kubectl symlink to k3s
[INFO] Creating /usr/local/bin/crictl symlink to k3s
[INFO] Creating /usr/local/bin/ctr symlink to k3s
[INFO] Creating killall script /usr/local/bin/k3s-killall.sh
[INFO] Creating uninstall script /usr/local/bin/k3s-agent-uninstall.sh
[INFO] env: Creating environment file /etc/systemd/system/k3s-agent.service.env
[INFO] systemd: Creating service file /etc/systemd/system/k3s-agent.service
[INFO] systemd: Enabling k3s-agent unit
Created symlink /etc/systemd/system/multi-user.target.wants/k3s-agent.service → /etc/systemd/system/k
.
[INFO] systemd: Starting k3s-agent

```

- Once the new device sets up all the required dependencies on itself and joins the cluster, it shows up on the Device Management Dashboard under the “More Devices in the Cluster” section.

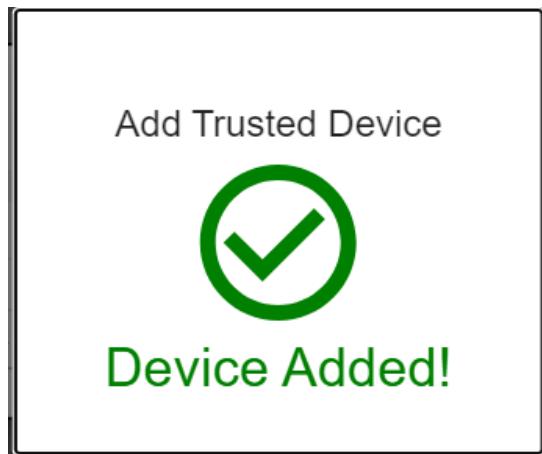
- As we would need a surveillance drone, a supply drone, a relay drone and a controller, add two more devices to the cluster using the same procedure as above.

11. On clicking on an device from the “More Devices” section, we can configure them further as trusted devices to attain virtualized behavior by deploying containers on them or remove the devices from the cluster.



12. By configuring the devices as trusted devices, we can deploy them as certain types of drones/rovers/devices with respective functionalities, and adjust the allowed operations for the devices, which dictate the ingress and egress rules for the devices.

Device Name *	surveillancedrone
IP Address *	172.31.47.238
The device will be assigned a local IP address after setup and will no longer be referenced by the public IP address.	
Device Type *	Video Capture Drone
Allowed Operations	
<input type="button" value="send_video"/> <input type="button" value="send_sensordata"/> <input type="button" value="receive_command"/>	
Possible Ingress Rules *	
5025/TCP	
Egress Rules *	
5005/UDP 5035/TCP	



13. Configure all four devices required for the mission execution with their respective drone types and capabilities as shown in the configuration screenshots below.

The screenshot shows the Device Management Dashboard. On the left, a sidebar menu includes "Manage Edge Devices" which is currently selected. The main area displays a grid titled "Configured Devices" containing icons for a surveillance drone, a controller, a relay drone, and a supply drone. Below the grid is a link "More Devices in the Cluster".

Four separate "Configured Device Information" pop-up windows are shown, each corresponding to one of the devices listed in the dashboard:

- surveillancedrone**: IP Address 10.42.1.3, Device Type Video Capture Drone. Allowed Operations: send_video, send_sensordata, send_posdata, receive_command.
- controller**: IP Address 10.42.0.9, Device Type Video Analytic Controller. Allowed Operations: send_command, receive_posdata, receive_sensordata, receive_video.
- relaydrone**: IP Address 10.42.3.3, Device Type Communication Relay Drone. Allowed Operations: send_command, send_video, send_sensordata, send_posdata, receive_posdata, receive_sensordata, receive_video, receive_command.
- supplydrone**: IP Address 10.42.2.3, Device Type Freight Drone. Allowed Operations: send_sensordata, send_posdata, receive_command.

IV. Deploying, Managing and Monitoring Honeypots

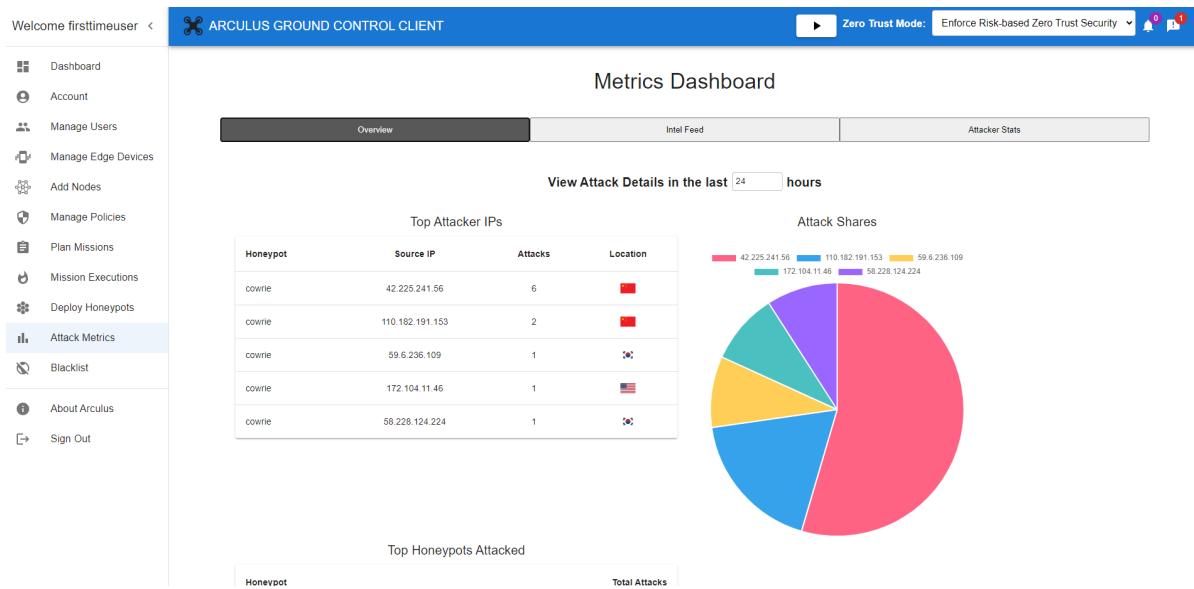
1. The Honeypot dashboard of the Arculus web interface allows for the seamless deployment of multiple types of honeypots. The target device and the honeypot can be selected based on the type of vulnerabilities to be simulated for active defense using the capabilities of the CHN Server.

The screenshot shows the Honeypot Dashboard page. On the left is a sidebar with navigation links: Dashboard, Account, Manage Users, Manage Edge Devices, Add Nodes, Manage Policies, Plan Missions, Mission Executions, Deploy Honeypots (which is highlighted), Attack Metrics, Blacklist, About Arculus, and Sign Out. The main content area has a title "Honeypot Dashboard". Below it is a "Deploy New Honeypot" section with fields for "Device IP Address" (set to "172.31.45.90 (relaydrone)") and "Honeypot Type" (set to "Cowrie"). A "Honeypot Description" box contains text about Cowrie's behavior and attack strategy. At the bottom is a "DEPLOY HONEY POT" button.

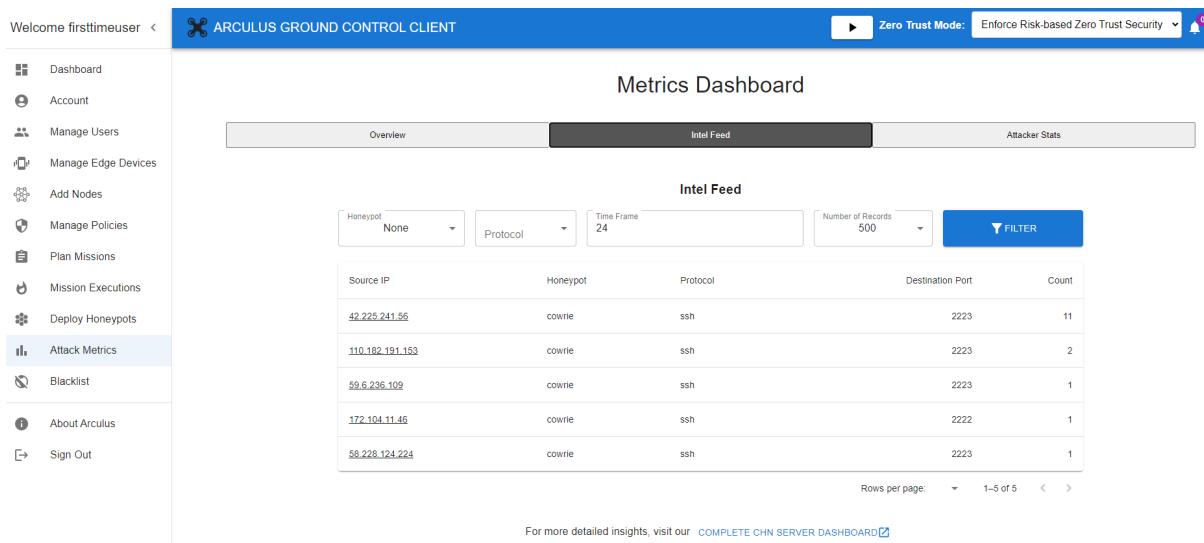
2. The deployed honeypots can be viewed or deleted on the “Deployed HoneyPots” pf the Honeypot dashboard.

The screenshot shows the Honeypot Dashboard page with the "Deployed HoneyPots" tab selected in the top navigation bar. The main content area displays a table titled "Deployed HoneyPots" with one row. The table has columns for "IP Address" (172.31.45.90), "Honeypot Type" (Cowrie), and "Actions" (which includes a circular icon and a red X). At the bottom is a "Powered by CommunityHoneyNet (CHN)" link.

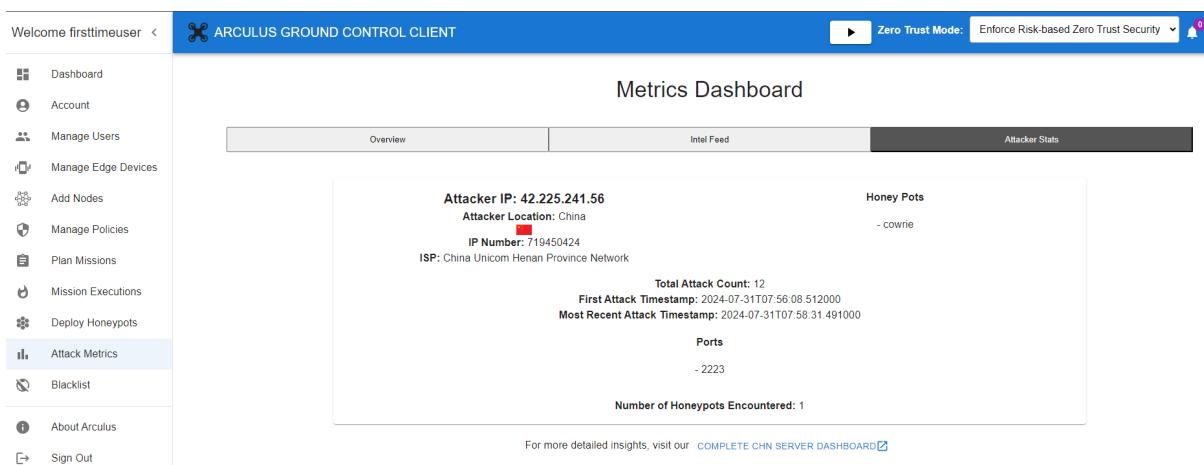
3. On the Metrics Dashboard, we can see the overview of all the attackers and their share of the total malicious activity on our sensors.



4. The Intel Feed gives deep and technical insights into what types of attacks using which protocols and ports were executed on the cluster. This gives us information on the strategies attackers are taking as a pattern to establish attack signatures.



5. We can also view the “Attacker Stats” tab where a detailed information of the source of attacks and the total count of the attack attempts can be seen. These dashboards provide a lot of threat intelligence which is also backed up on the CHN Server for detailed analysis.

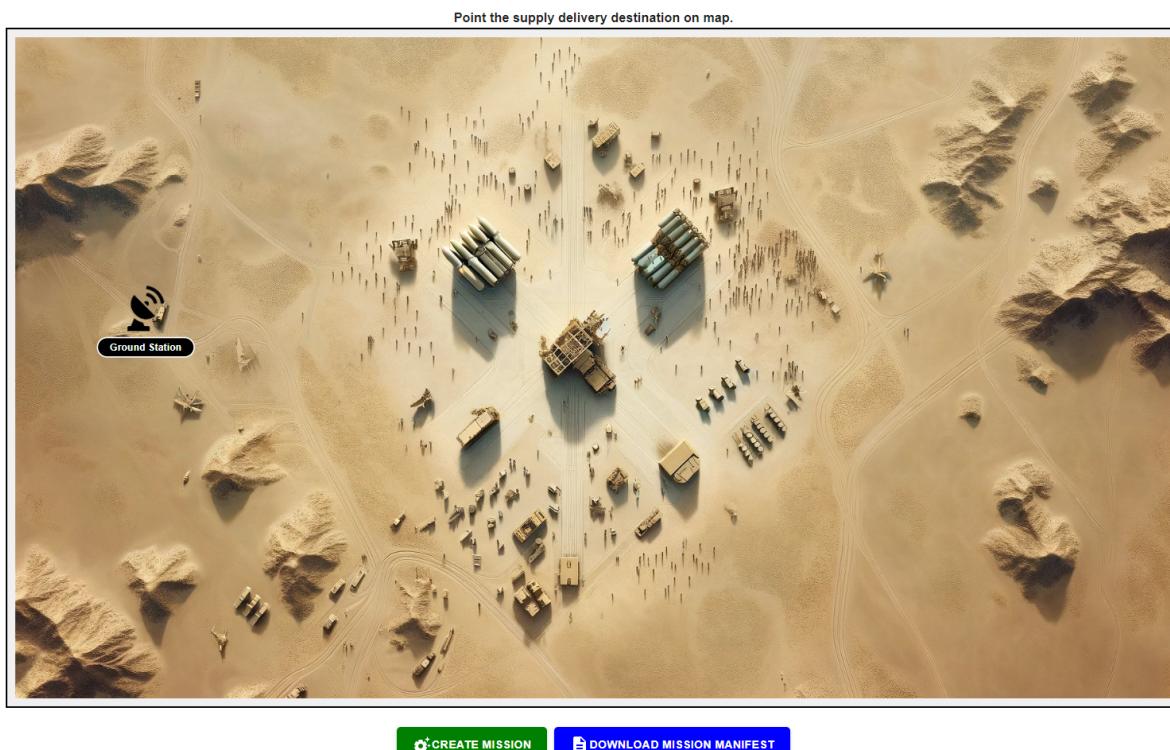


V. Planning and Execution of Drone Missions on the Arculus Testbed

- The Mission Planning Dashboard allows for configuring drone missions by specifying which devices to use, what the context (criticality) of the mission is, who the supervisors and viewers of the mission are, and logistics information. Configure these options to tune the criticality of the mission dictating the zero trust levels. The device fields are automatically populated based on the availability of the suitable types of devices.

The screenshot shows the 'Mission Planning Dashboard' interface. On the left is a sidebar with navigation links: Dashboard, Account, Manage Users, Manage Edge Devices, Add Nodes, Manage Policies, Plan Missions (selected), Mission Executions, Deploy Honeypots, Attack Metrics, Blacklist, About Arculus, and Sign Out. At the top right are 'Zero Trust Mode' settings ('Enforce Risk-based Zero Trust Security'), a notification bell with 0 notifications, and social sharing icons. The main area has tabs 'Select Mission Type' and 'Plan Mission' (selected). Below is the 'Plan Mission' configuration section. It includes fields for 'Mission Location' (set to 'Battlefield 1: Paleo Forest') and 'Video-Analytic Route Planner (Ground Control)' (set to 'controller'). Other fields include 'Video Collection Surveillance Drone' (set to 'surveillancedrone'), 'Supply Delivery Drone' (set to 'supplydrone'), 'Communication Relay Drone' (set to 'relaydrone'), 'Asset Criticality' (set to 'Low'), 'Life Threat' (set to 'Low'), 'Data Sensitivity' (set to 'Low'), 'Strategic Importance' (set to 'Low'), and 'Supervisors' (set to 'testuser'). A 'Viewers' field is also present. The overall theme is blue and white.

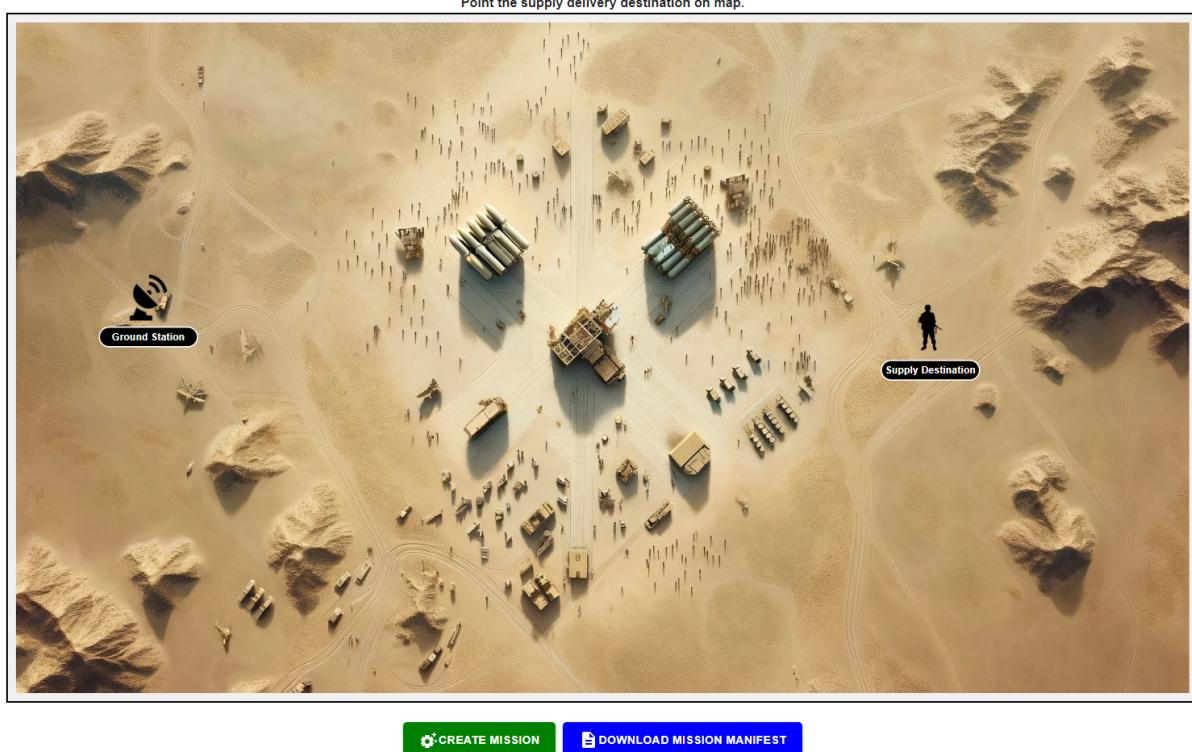
- The page also provides maps of locations to choose from. On these maps, users can pick a destination point for the delivery of supplies (here, the case of Stealthy Reconnaissance and Resupply mission). Try creating the mission without pointing a destination on the map, and the mission creation will not go ahead asking the user to place a destination on the map.



Please point the destination on the map!

OK

- Click anywhere on the map to set the supply destination to that location.



- Try updating the capabilities and remove one of the privileges specified in the device configuration step. The mission creation won't go ahead because, inability of any of the devices to perform their respective functions will cause the mission to fail for sure. This validation mechanism is a proactive approach to missions whose failure cannot be afforded.

Edit Trusted Device

Device Name* surveillancedrone

IP Address* 10.42.1.3

Allowed Operations
send_video send_sensordata
receive_command send_posdata

Possible Ingress Rules*
5005/UDP
5015/TCP
5035/TCP

Egress Rules*
5025/TCP

UPDATE TRUSTED DEVICE

CANCEL

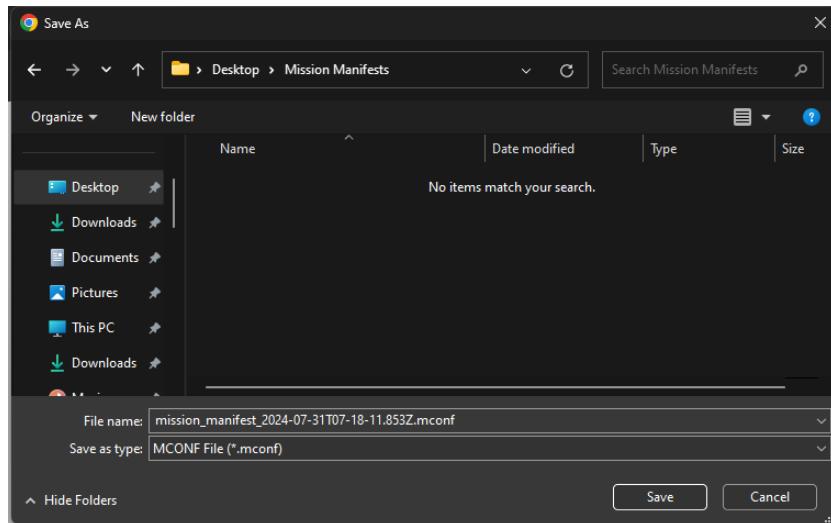
Insufficient Privileges for Some Devices

Mission Execution might fail. Please provide sufficient capabilities for necessary ingress and egress.

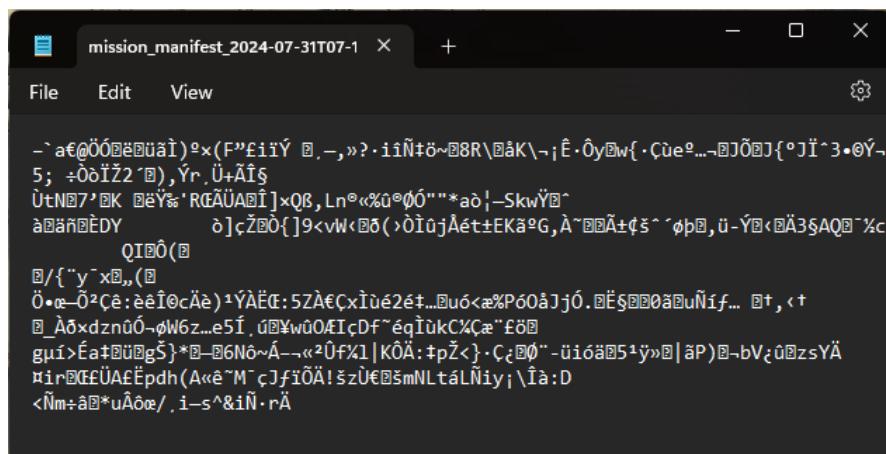
Device	Privileges
controller	<input checked="" type="checkbox"/> send_command <input checked="" type="checkbox"/> receive_video <input checked="" type="checkbox"/> receive_posdata
surveillancedrone	<input checked="" type="checkbox"/> send_video <input checked="" type="checkbox"/> receive_command <input type="checkbox"/> send_posdata
supplydrone	<input checked="" type="checkbox"/> receive_command <input checked="" type="checkbox"/> send_posdata
relaydrone	<input checked="" type="checkbox"/> receive_command <input checked="" type="checkbox"/> send_command <input checked="" type="checkbox"/> send_video <input checked="" type="checkbox"/> send_posdata <input checked="" type="checkbox"/> receive_posdata <input checked="" type="checkbox"/> receive_video

OK

5. Besides normal creation of missions, the “Download Mission Manifest” button allows users to download a repeatable manifest of their mission. This prevents multiple reconfigurations of a mission. The files are encrypted and downloaded with a .mconf extension.



6. From the screenshot below, the manifest files are encrypted and hence, modifications to these files by external actors invalidates the files, leaving no space for deliberate mission adulteration. The next screenshot shows how the Arculus application fetches the mission information from the .mconf file on executing with manifest.



Mission Execution Dashboard

Missions created by you Execute using Mission Manifest File Mission Execution

Upload Manifest File for Mission Execution

[SELECT A DIFFERENT MANIFEST FILE](#) mission_manifest_2024-07-31T07-18-11.853Z.mconf [UPLOAD FILE](#)

Mission Configuration:

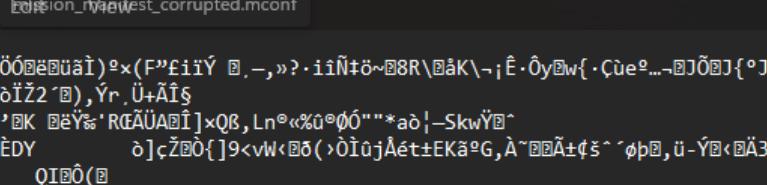
```

location: desert
mission_type: Stealthy Reconnaissance and Resupply
gcX: 202.3168
gcY: 420.0448
destX: 1375.8081239074452
destY: 387.01468017518465
selections:
Video-Analytic Route Planner (Ground Control): controller
Video Collection Surveillance Drone: surveillancedrone
Supply Delivery Drone: supplydrone
Communication Relay Drone: relaydrone
duration_sec: 120
Supervisor IDs:
Viewer IDs:

```

[EXECUTE MISSION](#)

7. However, try adding your own text to the manifest file and try to poison it as shown below. The application throws an error message identifying the manifest file to be bad.



The screenshot shows a text editor window with the title "mission_manifest_corrupted.mcon". The file content is severely corrupted, appearing as a series of random characters and symbols. The first few lines are partially readable as "mission_manifest_corrupted.mconf" and "File", while the rest of the file is completely garbled.

Mission Execution Dashboard

Missions created by you Execute using Mission Manifest File Mission Execution

Upload Manifest File for Mission Execution

[SELECT A DIFFERENT MANIFEST FILE](#)

mission_manifest_corrupted.mconf [X](#)

[UPLOAD FILE](#)

Bad File! Please choose an actual .mconf file.

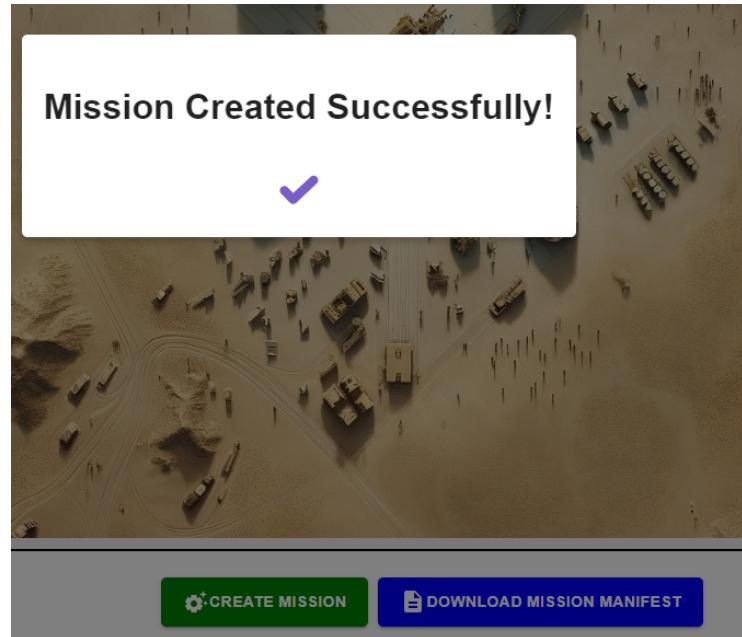
Mission Configuration:
Waiting for file upload...

[EXECUTE MISSION](#)

- Before trying to execute a mission, click the “Manage Policies” tab that takes you to the Network Policy Management Dashboard. It can be seen as in the below screenshot, that there are no active network policies and all traffic is denied by default.

No Active Network Policies (Deny By Default)

- Now, let us create a mission on the mission planning dashboard and execute it using the EXECUTE (>) button on the “Missions created by me” tab. The mission starts executing within seconds and the application takes us to the “Mission Execution” tab where the live mission and its logs can be monitored.



Mission Execution Dashboard

Missions created by you			Execute using Mission Manifest File			Mission Execution					
Mission Location	Mission Type	Creator	Supervisors	Viewers	Creation Time	Duration	Execute	Monitor	Status	Delete	
Grand Senora Desert	Stealthy Reconnaissance and Resupply	User admin			2024-07-31 07:19:27	120 sec	▶	🕒	CREATED	trash	

Mission Execution Dashboard

Missions created by you Execute using Mission Manifest File Mission Execution

Simulate Communication Loss Simulate GPS Spoofing Simulate Physical Capture Simulate Denial of Service Simulate Low Battery Simulate Brute Force SSH

Activity Log

Control Remotely

Surveillance Drone

Abort Mission

10. It can be observed that, on navigating to the Network Policy Management Dashboard, new policies are created during the mission execution to facilitate communication for respective tasks. However, as soon as the mission execution finishes/terminates, all the network policies are deleted. This demonstrates the extraordinary and dynamic execution of Task-based Access Control (TBAC) for good privilege management for devices on the network.

	Dashboard
	Account
	Manage Users
	Manage Edge Devices
	Add Nodes
	Manage Policies
	Plan Missions
	Mission Executions
	Deploy Honeypots
	Attack Metrics
	Blacklist
	About Arculus
	Sign Out

Network Policy Management Dashboard

Active Network Policies

New Network Policy

Active Network Policies

Policy ID	Device	Ingress Rules	Egress Rules	Edit	Delete
policy-controller	controller	supplydrone: 6002/UDP surveillancedrone: 6001/UDP relaydrone: 6003/UDP	relaydrone: 5050/TCP surveillancedrone: 3050/TCP supplydrone: 4050/TCP		
policy-relaydrone	relaydrone	controller: 5050/TCP	controller: 6003/UDP		
policy-supplydrone	supplydrone	controller: 4050/TCP relaydrone: 4050/TCP	controller: 6002/UDP		
policy-surveillancedrone	surveillancedrone	controller: 3050/TCP relaydrone: 3050/TCP	controller: 6001/UDP		

11. By playing around with the simulation buttons provided on the Mission Execution Dashboard, we can explore the various zero trust capabilities and countermeasures taken by Arculus for secure and successful completion of missions under adverse conditions. These include DDIL scenarios like communication loss, GPS Spoofing attacks, physical hijacking, Denial of Service, Low battery, Brute force attacks, etc.

Mission Execution Dashboard

Missions created by you
Execute using Mission Manifest File
Mission Execution

[Simulate Communication Loss](#)
[Simulate GPS Spoofing](#)
[Simulate Physical Capture](#)
[Simulate Denial of Service](#)
[Simulate Low Battery](#)
[Simulate Brute Force SSH](#)

Ground Control Station: 282.00, 420.00
 Surveillance Drone: 1005.51, 305.07
 Supply Drone: 262.32, 420.04
 Destination: 1375.81, 387.01
 Enemy Air Defense: Unknown

Activity Log
 Unknown move commands received. Authentication Failed.
 Potential GPS Spoofing.
 Closing Drone Ingress and taking pre-planned flight path.
 Communication failure with the drone.
 Sending out Relay Drone.
 Communication with Surveillance Drone Established.
 Communication with Surveillance Drone Established.

Control Remotely

Surveillance Drone

▲ ▼ ⌂ ⌃ ⌁ ⌂ ⌃ ⌁

Abort Mission

Mission Execution Dashboard

Missions created by you
Execute using Mission Manifest File
Mission Execution

[Simulate Communication Loss](#)
[Simulate GPS Spoofing](#)
[Simulate Physical Capture](#)
[Simulate Denial of Service](#)
[Simulate Low Battery](#)
[Simulate Brute Force SSH](#)

Ground Control Station: 282.00, 420.00
 Surveillance Drone: 206.02, 419.99
 Supply Drone: 1361.86, 307.41
 Destination: 1375.81, 387.01
 Enemy Air Defense: Unknown

Activity Log
 Communication with Surveillance Drone Established.
 Communication with Surveillance Drone Established.

Control Remotely

Surveillance Drone

▲ ▼ ⌂ ⌃ ⌁ ⌂ ⌃ ⌁

Abort Mission