



Blockchain Based Ticket Management

Capstone Intermediate Milestone Submission

Blockchain Group 1 (blockchaing1bbtm)

Ankit Shah
Kalaivanan S
Mohammed Shuaib Mohiuddin
Preeti Aggarwal

Apr 24, 2022



Overview

Develop a unique blockchain based ticket management system to entice users with transparency, and automated refunds and delay penalties for one Eagle Airlines.

Set-up a private Ethereum blockchain and develop a base ticketing contract, that would allow immediate refund in case of cancellations and predefined penalty payment in case of delays.



Scope

- 
- 01 Infrastructure - EC2 instances (2) to run Geth-CliquePoA nodes of the Private Blockchain network that hosts the Airline Ticket Management system and related Airline & Customer Accounts
 - 02 Core Application - Solidity contracts to implement the Ticket Management System's interaction with Airline & Customers.
 - 03 Advanced Features
Infra - Hyperledger Besu nodes with Tessera to mark transactions as private.
Application - Implement Oracle to fetch Flight details. Multiple cancellation & penalty scenarios. If possible, dApp creation using Truffle - one for Airline and one for Customers.



Private Blockchain Network Setup

01 Infrastructure

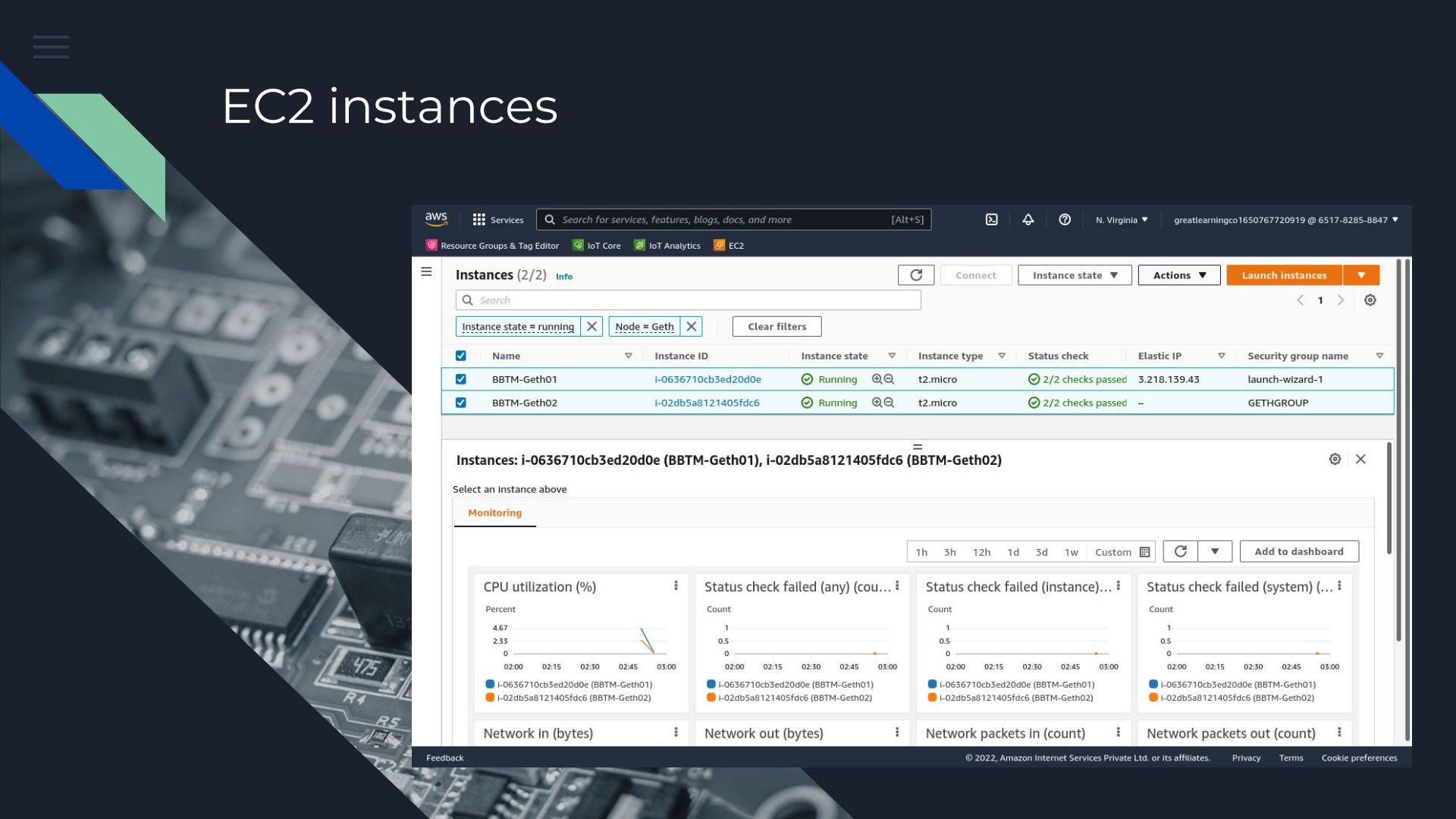


EC2 instances
2 x t2.micro

Blockchain Ethereum Clique
- 2 geth instances (one in each EC2) with 3 nodes
Instance-1 - 1 bootnode + 1 peer node
Instance-2 - 1 miner node

BC Accounts
2 Airline Accounts + 4 Passenger Accounts

EC2 instances



The screenshot shows the AWS Management Console interface for managing EC2 instances. The top navigation bar includes the AWS logo, a search bar, and links for Resource Groups & Tag Editor, IoT Core, IoT Analytics, and EC2. The main content area displays a table of running EC2 instances.

Instances (2/2) Info

| Name | Instance ID | Instance state | Instance type | Status check | Elastic IP | Security group name |
|-------------|---------------------|----------------|---------------|-------------------|--------------|---------------------|
| BBTM-Geth01 | i-0636710cb3ed20d0e | Running | t2.micro | 2/2 checks passed | 3.218.139.43 | launch-wizard-1 |
| BBTM-Geth02 | i-02db5a8121405fdc6 | Running | t2.micro | 2/2 checks passed | - | GETHGROUP |

Instances: i-0636710cb3ed20d0e (BBTM-Geth01), i-02db5a8121405fdc6 (BBTM-Geth02)

Select an instance above

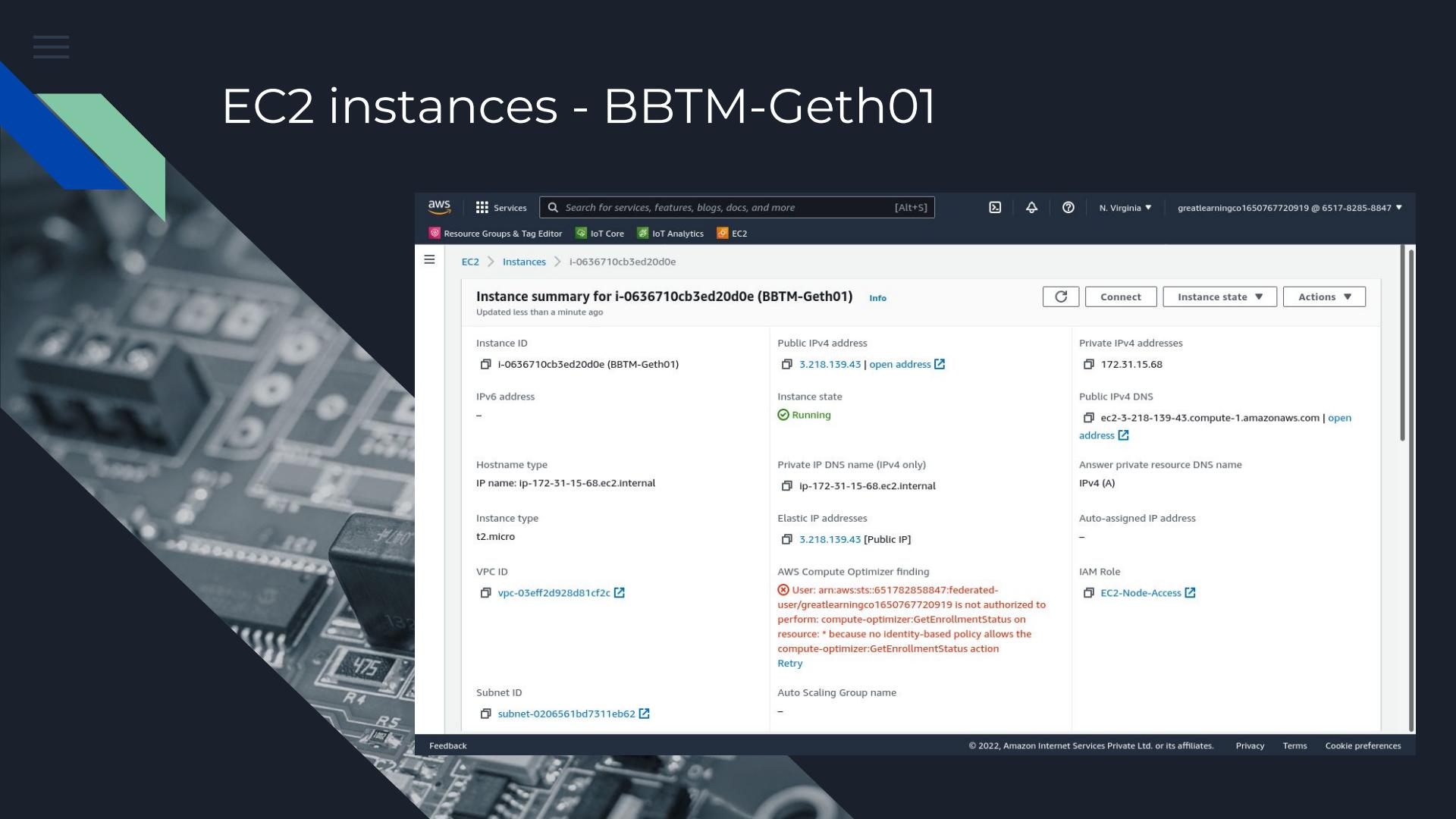
Monitoring

1h 3h 12h 1d 3d 1w Custom Add to dashboard

| CPU utilization (%) | Status check failed (any) (cou...) | Status check failed (instance)... | Status check failed (system) (...) |
|---|---|---|---|
| Percent 4.67 2.33 0 02:00 02:15 02:30 02:45 03:00 i-0636710cb3ed20d0e (BBTM-Geth01) i-02db5a8121405fdc6 (BBTM-Geth02) | Count 1 0.5 0 02:00 02:15 02:30 02:45 03:00 i-0636710cb3ed20d0e (BBTM-Geth01) i-02db5a8121405fdc6 (BBTM-Geth02) | Count 1 0.5 0 02:00 02:15 02:30 02:45 03:00 i-0636710cb3ed20d0e (BBTM-Geth01) i-02db5a8121405fdc6 (BBTM-Geth02) | Count 1 0.5 0 02:00 02:15 02:30 02:45 03:00 i-0636710cb3ed20d0e (BBTM-Geth01) i-02db5a8121405fdc6 (BBTM-Geth02) |
| Network in (bytes) | Network out (bytes) | Network packets in (count) | Network packets out (count) |

Feedback © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

EC2 instances - BBTM-Geth01



The screenshot shows the AWS Management Console interface for an EC2 instance named "BBTM-Geth01". The instance is currently running and has a public IPv4 address of 3.218.139.43. It is associated with a VPC ID (vpc-03eff2d928d81cf2c) and a subnet ID (subnet-0206561bd7311eb62). The instance type is t2.micro. The IAM role assigned to this instance is EC2-Node-Access. The instance summary also includes details like the private IP DNS name (ip-172-31-15-68.ec2.internal), elastic IP addresses (3.218.139.43), and the AWS Compute Optimizer finding, which indicates a user lacks permission to perform certain actions.

aws Services Search for services, features, blogs, docs, and more [Alt+S] N. Virginia greatlearningco1650767720919 @ 6517-8285-8847

Resource Groups & Tag Editor IoT Core IoT Analytics EC2

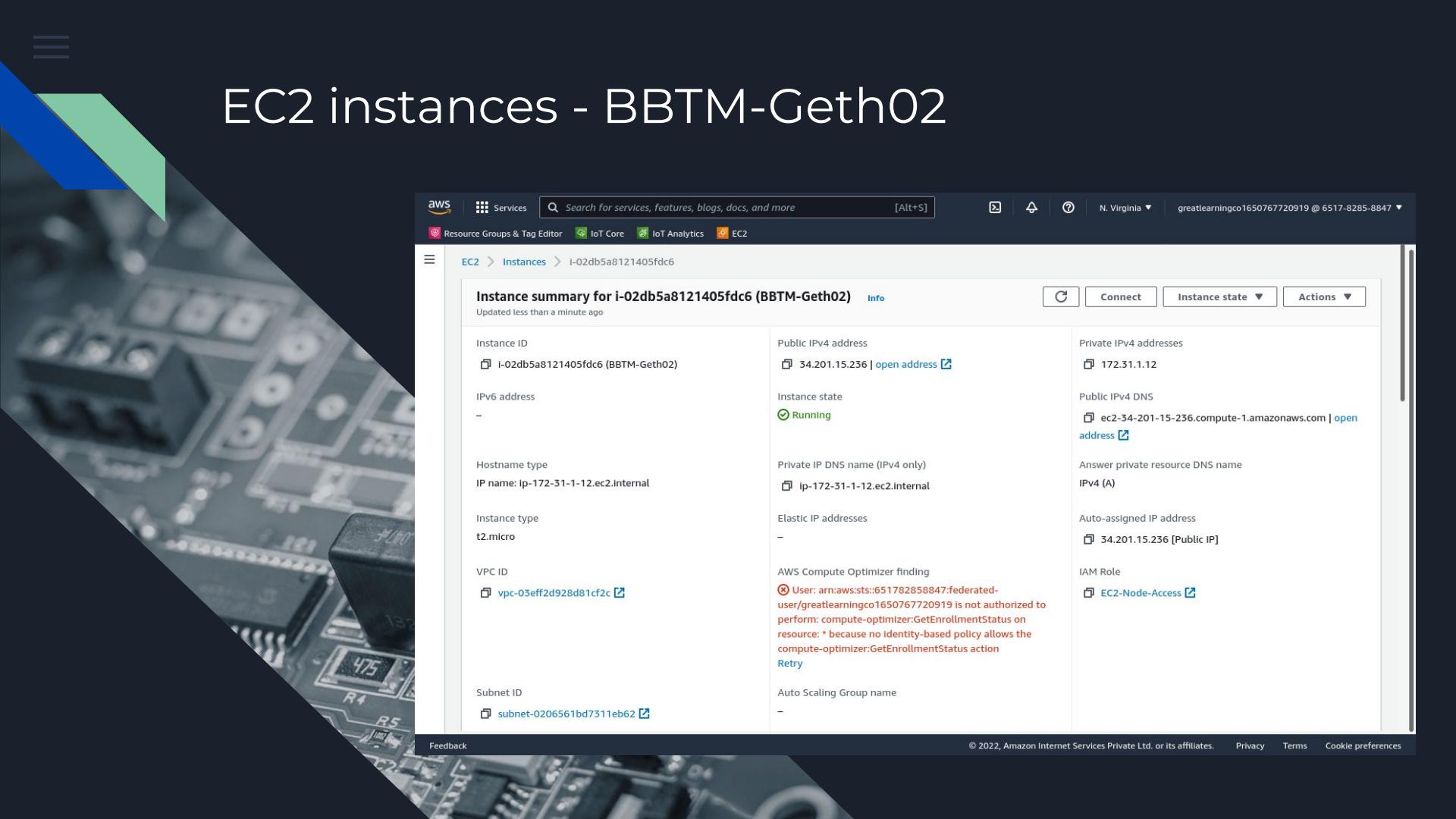
EC2 Instances i-0636710cb3ed20d0e

Instance summary for i-0636710cb3ed20d0e (BBTM-Geth01) Info Updated less than a minute ago

| Instance ID | Public IPv4 address | Private IPv4 addresses |
|---------------------------------------|--|---|
| i-0636710cb3ed20d0e (BBTM-Geth01) | 3.218.139.43 open address | 172.31.15.68 |
| IPv6 address | Instance state | Public IPv4 DNS |
| - | Running | ec2-3-218-139-43.compute-1.amazonaws.com open address |
| Hostname type | Private IP DNS name (IPv4 only) | Answer private resource DNS name |
| IP name: ip-172-31-15-68.ec2.internal | ip-172-31-15-68.ec2.internal | IPv4 (A) |
| Instance type | Elastic IP addresses | Auto-assigned IP address |
| t2.micro | 3.218.139.43 [Public IP] | - |
| VPC ID | AWS Compute Optimizer finding | IAM Role |
| vpc-03eff2d928d81cf2c | User: arn:aws:sts::65178285847:federated-user/greatlearningco1650767720919 is not authorized to perform: compute-optimizer:GetEnrollmentStatus on resource: * because no identity-based policy allows the compute-optimizer:GetEnrollmentStatus action | EC2-Node-Access |
| Subnet ID | Auto Scaling Group name | |
| subnet-0206561bd7311eb62 | - | |

Feedback © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

EC2 instances - BBTM-Geth02



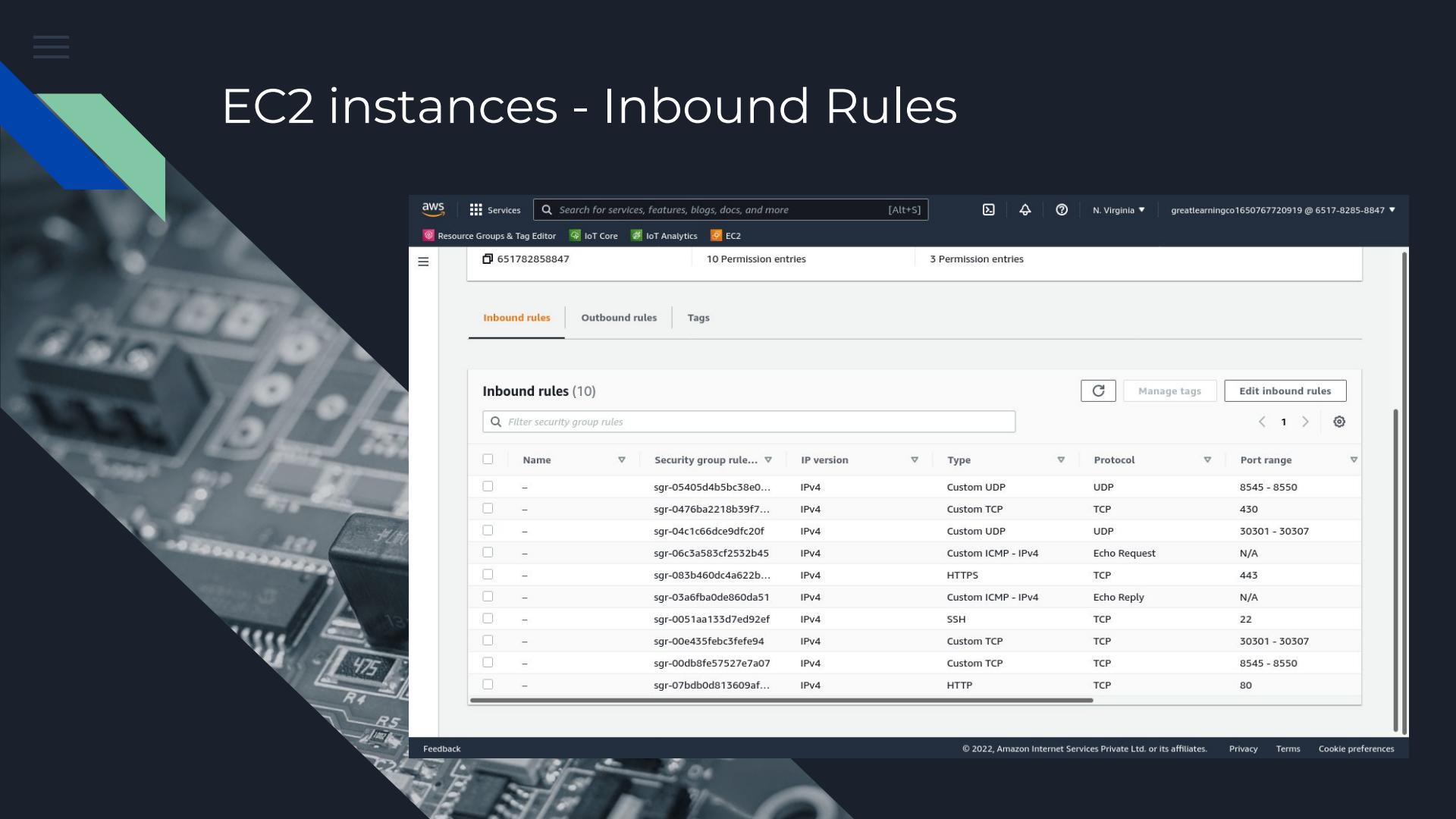
The screenshot shows the AWS Management Console interface for an EC2 instance named "BBTM-Geth02". The instance is currently running and has a public IPv4 address of 34.201.15.236. It is associated with a VPC ID of vpc-03eff2d928d81cf2c and a subnet ID of subnet-0206561bd7311eb62. The instance type is t2.micro. The IAM role assigned to this instance is EC2-Node-Access.

Instance summary for i-02db5a8121405fdc6 (BBTM-Geth02)

Updated less than a minute ago

| Attribute | Value | Actions |
|----------------------------------|--|----------------|
| Instance ID | i-02db5a8121405fdc6 (BBTM-Geth02) | Copy Connect |
| IPv6 address | - | |
| Hostname type | IP name: ip-172-31-1-12.ec2.internal | |
| Instance type | t2.micro | |
| VPC ID | vpc-03eff2d928d81cf2c | |
| Subnet ID | subnet-0206561bd7311eb62 | |
| Public IPv4 address | 34.201.15.236 open address | |
| Private IP DNS name (IPv4 only) | ip-172-31-1-12.ec2.internal | |
| Elastic IP addresses | - | |
| AWS Compute Optimizer finding | User: arn:aws:sts::65178285847:federated-user/greatlearningco1650767720919 is not authorized to perform: compute-optimizer:GetEnrollmentStatus on resource: * because no identity-based policy allows the compute-optimizer:GetEnrollmentStatus action | |
| Auto Scaling Group name | - | |
| Private IPv4 addresses | 172.31.1.12 | |
| Public IPv4 DNS | ec2-34-201-15-236.compute-1.amazonaws.com open address | |
| Answer private resource DNS name | IPv4 (A) | |
| Auto-assigned IP address | 34.201.15.236 [Public IP] | |
| IAM Role | EC2-Node-Access | |

EC2 instances - Inbound Rules



Screenshot of the AWS Management Console showing the Inbound rules for a security group (sg-05405d4b5bc38e0...). The interface includes a search bar, navigation links for Resource Groups & Tag Editor, IoT Core, IoT Analytics, and EC2, and account information for N. Virginia and greatlearningco1650767720919 @ 6517-8285-8847.

The main view displays 10 Permission entries under the Inbound rules tab. A secondary panel shows 3 Permission entries under the Outbound rules tab.

| Name | Security group rule... | IP version | Type | Protocol | Port range |
|------|------------------------|------------|--------------------|--------------|---------------|
| - | sgr-05405d4b5bc38e0... | IPv4 | Custom UDP | UDP | 8545 - 8550 |
| - | sgr-0476ba2218b39f7... | IPv4 | Custom TCP | TCP | 430 |
| - | sgr-04c1c66dce9dfc20f | IPv4 | Custom UDP | UDP | 30301 - 30307 |
| - | sgr-06c3a583cf2532b45 | IPv4 | Custom ICMP - IPv4 | Echo Request | N/A |
| - | sgr-083b460dc4a622b... | IPv4 | HTTPS | TCP | 443 |
| - | sgr-03a6fba0de860da51 | IPv4 | Custom ICMP - IPv4 | Echo Reply | N/A |
| - | sgr-0051aa133d7ed92ef | IPv4 | SSH | TCP | 22 |
| - | sgr-00e435febcb3fe94 | IPv4 | Custom TCP | TCP | 30301 - 30307 |
| - | sgr-00db8fe57527e7a07 | IPv4 | Custom TCP | TCP | 8545 - 8550 |
| - | sgr-07bdb0d813609af... | IPv4 | HTTP | TCP | 80 |

Feedback © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

EC2 instances - Geth - Boot Node

Activities Terminal ▾ Apr 24 08:58

ubuntu@ip-172-31-15-68: ~/gethpoa/bnode

```
ubuntu@ip-172-31-15-68:~:/gethpoa/bnode$ bootnode -nodekey ./boot.key -verbosity 7 -addr 172.31.15.68:30301  
enode://e8bf4c0a6930309e9d2eb3745893bf837179a6ec39fbfb9c670cc5593bd67e38305c69acf0ce0f0b7cdabbe785094d376011d9e8bf597640ab8212852e5a9d@172.31.15.68:  
0?discport=30301  
Note: you're using cmd/bootnode, a developer tool.  
We recommend using a regular node as bootstrap node for production deployments.
```

INFO [04-24|03:26:25.430] New local node record seq=1,650,770,785,427 id=6807a7d785b4796b ip=<nil> udp=0 tcp=0

TRACE[04-24|03:27:40.527] <> PING/v4
TRACE[04-24|03:27:40.527] >> PONG/v4
TRACE[04-24|03:27:40.528] <> PING/v4
TRACE[04-24|03:27:40.529] <> PING/v4
TRACE[04-24|03:27:40.529] >> PONG/v4
TRACE[04-24|03:27:40.529] >> PONG/v4
TRACE[04-24|03:27:40.544] <> PONG/v4
TRACE[04-24|03:27:40.544] <> PONG/v4
TRACE[04-24|03:27:41.040] <> FINDNODE/v4
TRACE[04-24|03:27:41.041] >> NEIGHBORS/v4
TRACE[04-24|03:27:41.060] <> FINDNODE/v4
TRACE[04-24|03:27:41.061] >> NEIGHBORS/v4
TRACE[04-24|03:27:41.542] <> FINDNODE/v4
TRACE[04-24|03:27:41.542] >> NEIGHBORS/v4
TRACE[04-24|03:27:41.567] <> FINDNODE/v4
TRACE[04-24|03:27:41.567] >> NEIGHBORS/v4
TRACE[04-24|03:27:42.044] <> FINDNODE/v4
TRACE[04-24|03:27:42.044] >> NEIGHBORS/v4
TRACE[04-24|03:27:42.067] <> FINDNODE/v4
TRACE[04-24|03:27:42.068] >> NEIGHBORS/v4
TRACE[04-24|03:27:42.545] <> FINDNODE/v4
TRACE[04-24|03:27:42.545] >> NEIGHBORS/v4
TRACE[04-24|03:27:42.569] <> FINDNODE/v4
TRACE[04-24|03:27:42.569] >> NEIGHBORS/v4
TRACE[04-24|03:27:43.070] <> FINDNODE/v4
TRACE[04-24|03:27:43.070] >> NEIGHBORS/v4
TRACE[04-24|03:27:43.571] <> FINDNODE/v4
TRACE[04-24|03:27:43.572] >> NEIGHBORS/v4
TRACE[04-24|03:27:44.074] <> FINDNODE/v4
TRACE[04-24|03:27:44.074] >> NEIGHBORS/v4

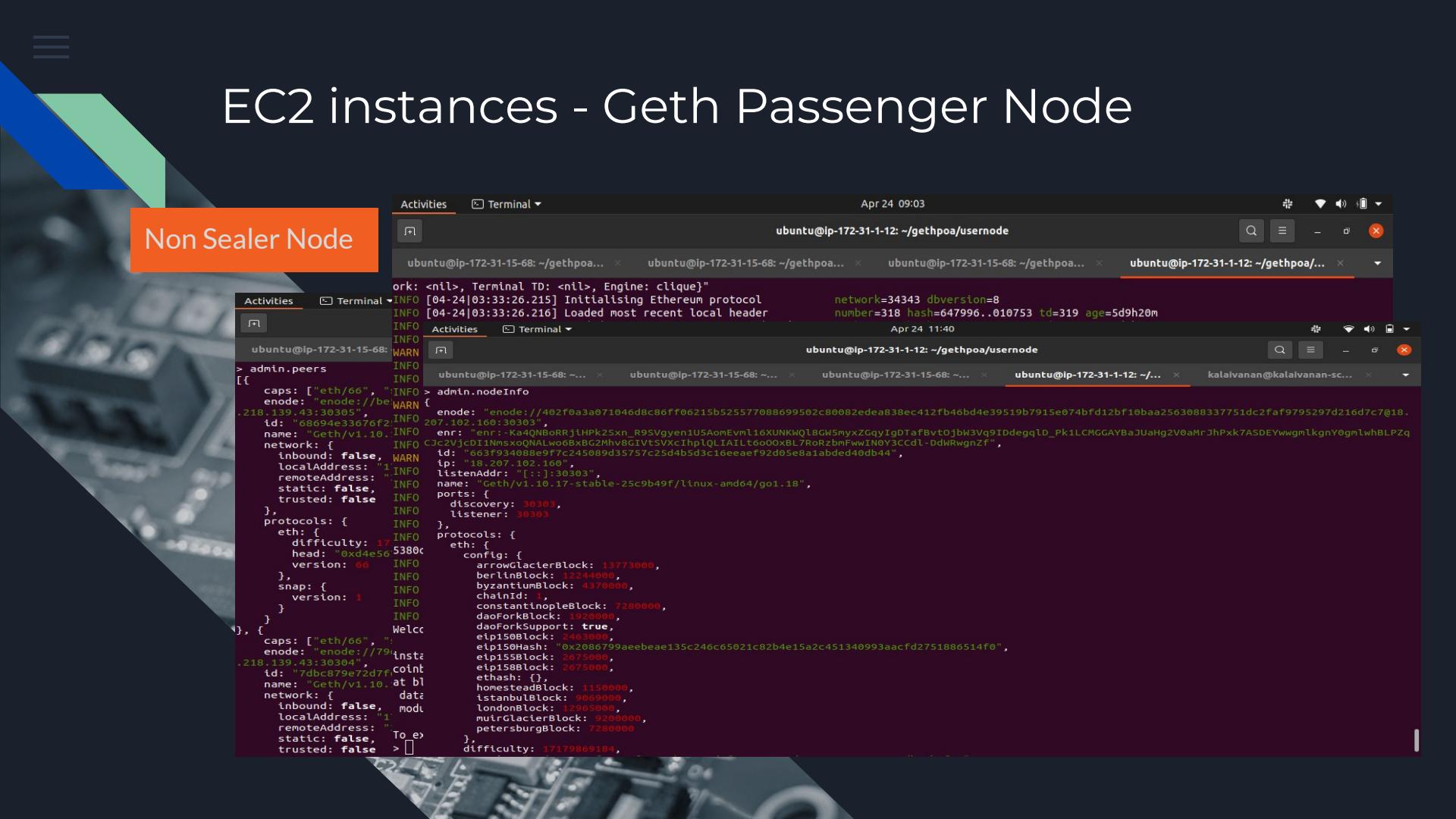
Activities Terminal ▾ Apr 24 11:38

ubuntu@ip-172-31-15-68: ~/gethpoa/bnode

```
ubuntu@ip-172-31-15-68:~:/gethpoa/bnode$  
u=68094e3501625db addr=3.218.139.43:30305 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=663f934088ef7c2 addr=172.31.1.12:30303 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=7dbc879e72d7fcc addr=3.218.139.43:30304 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil  
id=68694e33676f25db addr=3.218.139.43:30305 err=nil
```


EC2 instances - Geth Passenger Node

Non Sealer Node



A screenshot of a Linux desktop environment showing two terminal windows. The top terminal window is titled 'Activities Terminal' and shows a log from a Geth node at IP 172.31.1.12. The log output includes:

```
INFO [04-24|03:33:26.215] Initialising Ethereum protocol
INFO [04-24|03:33:26.216] Loaded most recent local header
INFO [04-24|03:33:26.216] network=34343 dbversion=8
INFO [04-24|03:33:26.216] number=318 hash=647996..010753 td=319 age=5d9h20m
INFO [04-24|03:33:26.216] Apr 24 11:40
```

The bottom terminal window is also titled 'Activities Terminal' and shows a log from a Geth node at IP 172.31.1.12. The log output includes:

```
INFO [04-24|03:33:26.215] Initialising Ethereum protocol
INFO [04-24|03:33:26.216] Loaded most recent local header
INFO [04-24|03:33:26.216] network=34343 dbversion=8
INFO [04-24|03:33:26.216] number=318 hash=647996..010753 td=319 age=5d9h20m
INFO [04-24|03:33:26.216] Apr 24 11:40
```

Both terminals show the command 'admin.peers' being run, displaying a list of connected nodes with their details such as IP address, port, and version.

Connecting BBTM - Private Blockchain Network via Metamask

The screenshot shows the Remix IDE interface. On the left, there's a sidebar with tabs for "DEPLOY & RUN TRANSACTIONS", "ENVIRONMENT" (set to "Injected Web3"), "ACCOUNT" (with a dropdown for "Custom (34343) network"), "GAS LIMIT" (set to 3000000), "VALUE" (set to 0 Wei), and "CONTRACT" (selected "IAirline - contracts/Capstone-Airline"). Below these are buttons for "Deploy" and "Publish to IPFS". The main area is titled "Remix IDE" and contains sections for "Featured Plugins" (Solidity, Starknet, Solfint Linter, Learneth), "File" (New File, Open Files, Connect to Localhost), and "Resources" (Documentation, Gitter channel, Featuring website). At the bottom, there's a search bar and a command line input field. A modal window titled "MetaMask Notification" is open, prompting the user to "Connect With MetaMask" and listing accounts: Account 1 (0x221...4040) (unchecked), BBTM-ATM01 (0x8dc...03...) (checked), BBTM-ATM02 (0xaca...97...) (unchecked), and BBTM-Passe... (0xa73...8...) (unchecked). The "Next" button is visible at the bottom right of the modal.

The screenshot shows the Metamask settings dialog. It has fields for "Network Name" (GL-Capstone-BBTM-G1), "New RPC URL" (http://3.218.139.43:8547), "Chain ID" (34343), "Currency Symbol" (ETH), and a "Block Explorer URL (Optional)" field which is empty. The "Network Name" field is highlighted with a red border.

| | |
|-------------------------------|--------------------------|
| Network Name | GL-Capstone-BBTM-G1 |
| New RPC URL | http://3.218.139.43:8547 |
| Chain ID | 34343 |
| Currency Symbol | ETH |
| Block Explorer URL (Optional) | |

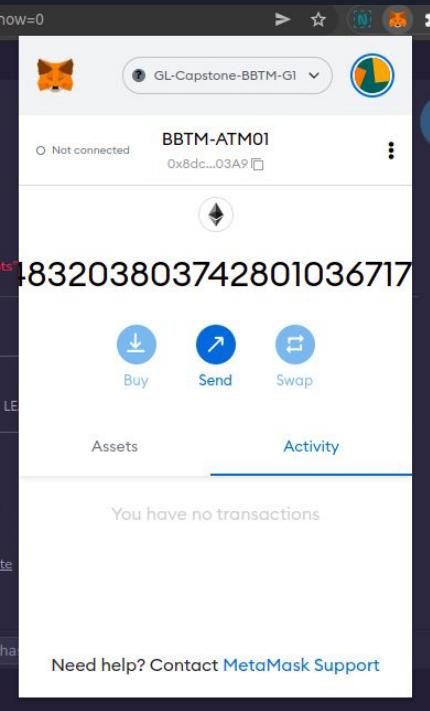
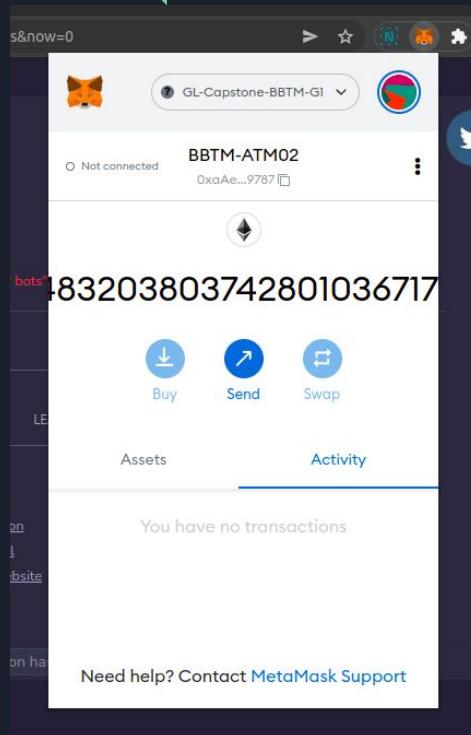
Airline Addresses

BBTM-ATM01

0x8dcf9c15834b5056BEa44F3446d860bE532903A9

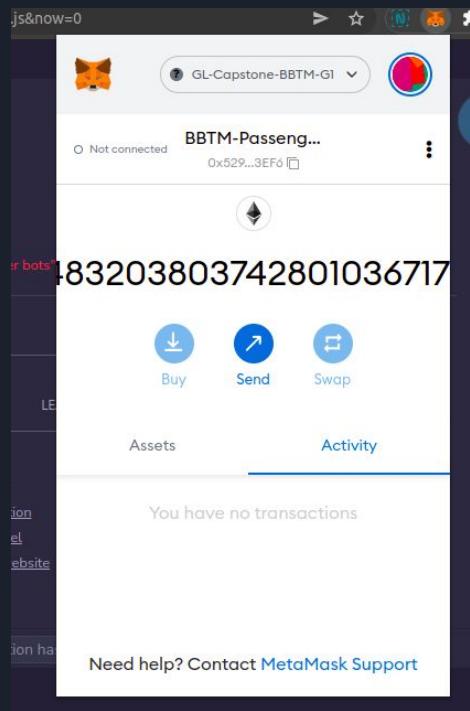
BBTM-ATM02

0xAeF6375968F1DA352F7Df62FD597b8B91739787



Passenger Addresses

BBTM-Passenger001 - 0xa7386bBA929DA248F83577723AeE5dEE7B128AD7
BBTM-Passenger002 - 0x6759bf46596B16e00450D9e965F33C0108fcfa65c
BBTM-Passenger003 - 0x529A5feEcB002B47B611bB32cEcCc944f1873EF6
BBTM-Passenger004 - 0x92FcF68DbAe5c4C08D30870Ea156772aC2B46b7B





Private Blockchain creation - Challenges & Learnings

1. Initially nodes running were not able to find any peers, as they other node was running on another ec2 instance. We used local host port 30301 to run boot node, And node on another EC2 kept searching for boot node on local 30301 port. USE of private ip on boot node along with 30301 port made possible for another node on ec2 to attach to that particular boot node and find peers connected to that bootnode.
2. Initially both sealer admin accounts were assigned to one node and due to that they were not able to mine block. **Running both miner accounts separately on different sealer node** made possible for them to mine blocks one at a time.
3. Initially metamask couldn't connect to chain, as there was no public IP assigned to node. By default 127.0.0.1 was assigned to each node. **Using "--nat extip", we overcome this hurdle.**
4. Even after assigning Public Ip seemed not working, and Use of METAMASK & injected web3 was not possible Initially as only rpc calls to local host only allowed. **HOWEVER** with use of flag "`--http.addr 0.0.0.0`", all rpc calls from anywhere were allowed and we could connect to chain using a valid rpc URL



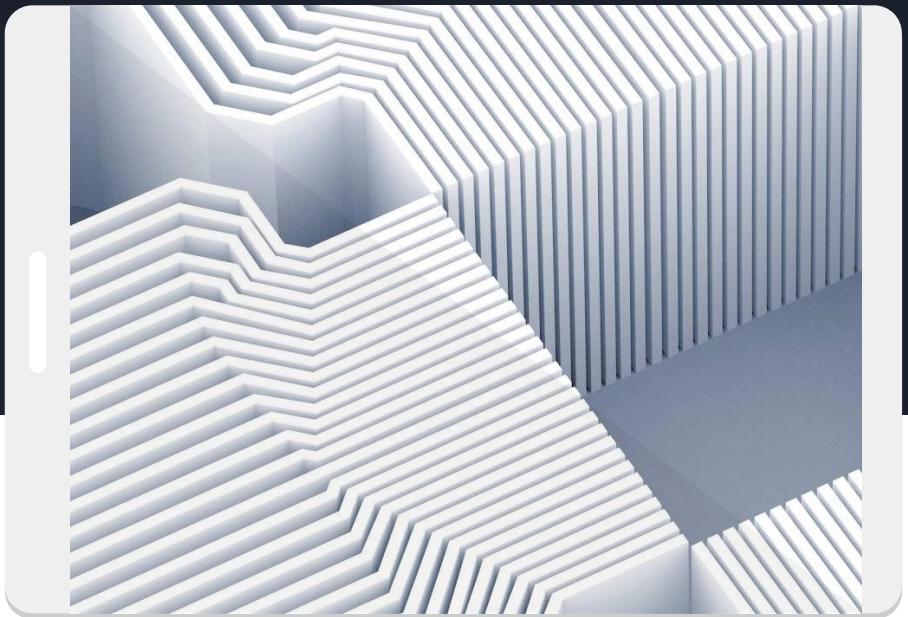
Base Contract for Airline Ticket Management

02 Core Application

Feature Summary

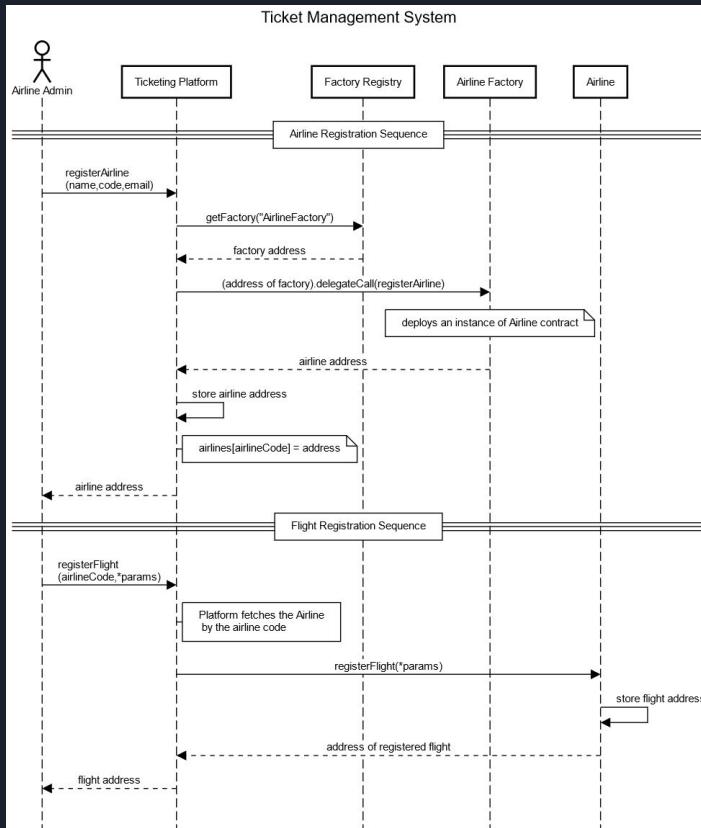
Register Airlines, Flights, Customers

List Available Seats, Book & Cancel Tickets,
Penalties & Refunds

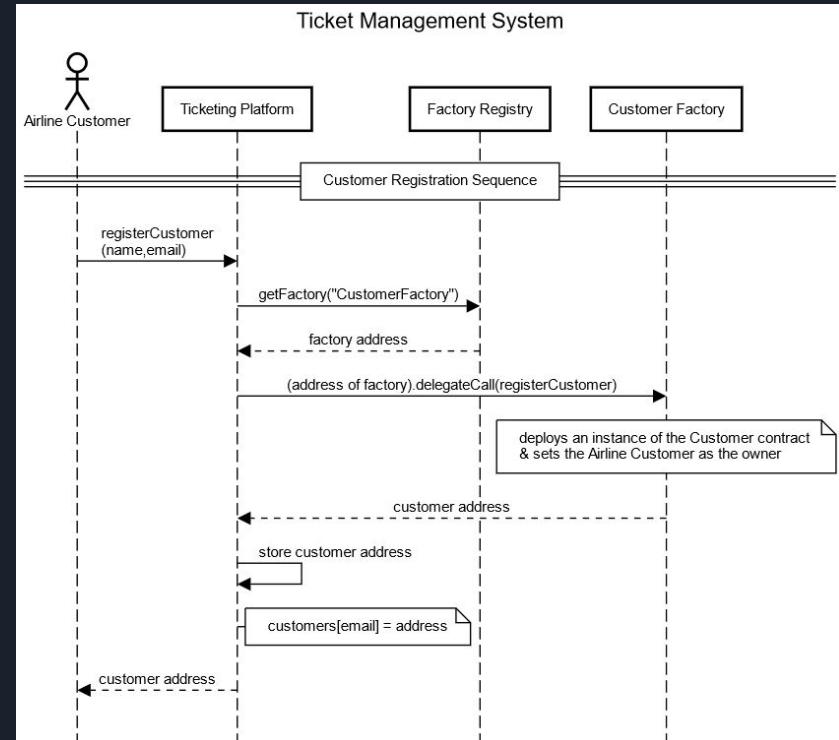


Contract Design - Registrations

Registering Airline & Flight

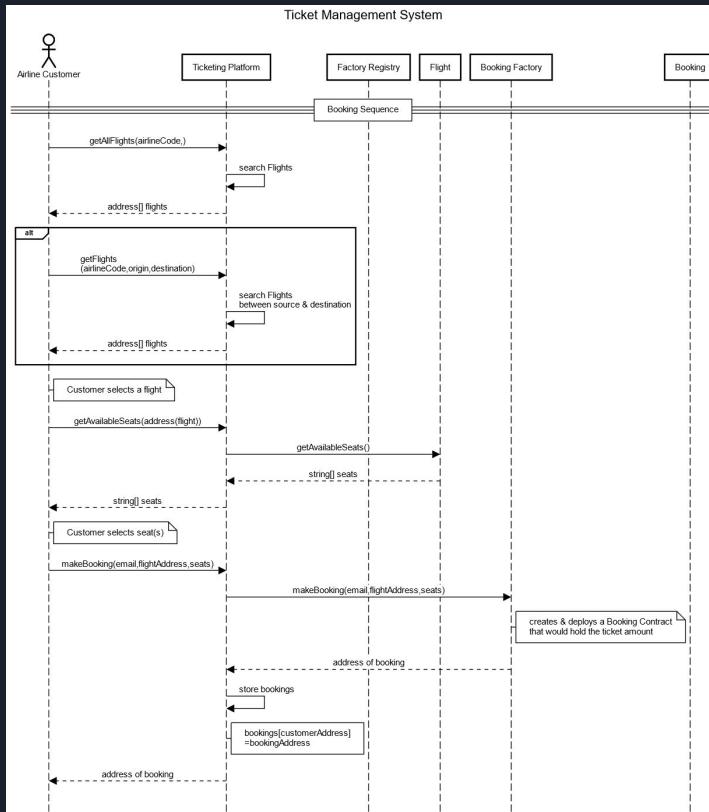


Registering a Customer

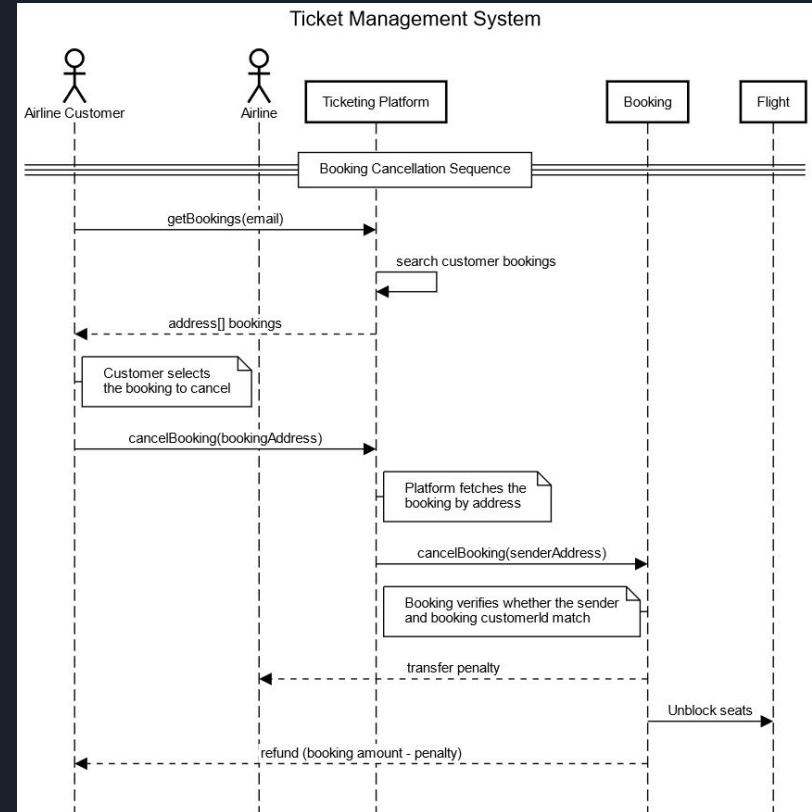


Contract Design - Customer Booking & Cancellation

Ticket Booking

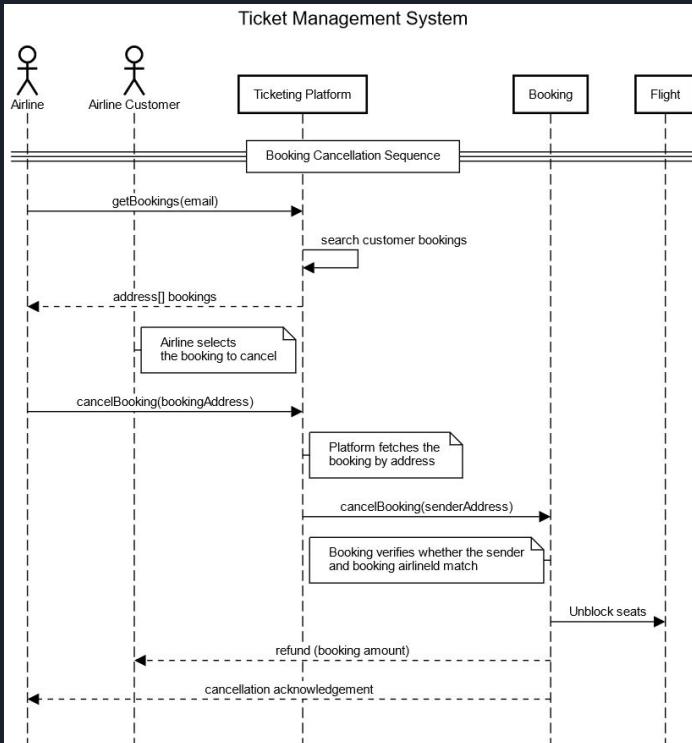


Ticket Cancellation by Customer

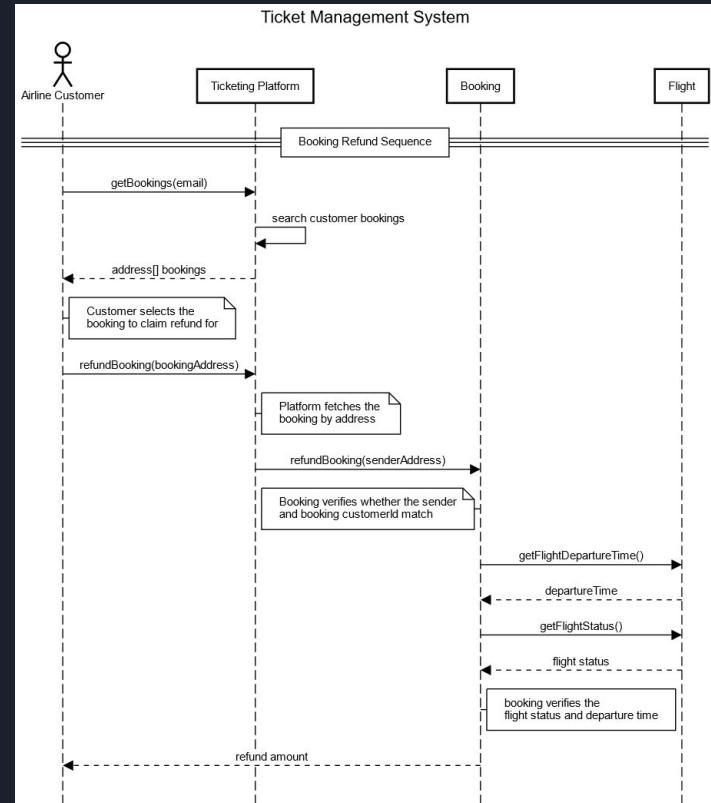


Contract Design - Airline Cancel & Claim Refund

Ticket Cancellation by Airline



Refund Claim by Customer





Initializing the BBTM on the Private Blockchain

1. Connect to one of the Airline Accounts (say, BBTM-ATM01) in Metamask and connect via InjectedWeb3 in Remix IDE
2. Deploy the below contracts* (no particular order)
 - o AirlineFactory
 - o BookingFactory
 - o CustomerFactory
 - o TicketingPlatform
3. Call the **getRegistryAddress** function of the deployed **TicketingPlatform** contract and copy the Address returned
4. Call the **updateRegistry** function of the Factory contracts - AirlineFactory, BookingFactory and CustomerFactory, filling in the copied Registry Address
5. System is ready for Ticket Management. Start by registering Airlines, Customers and Flights.

Contract Deployment

3. Once Sealed, the Contract is available in Deployed Contracts.

1. Import Contract Files

FILE EXPLORERS

Workspaces

default_workspace

contracts artifacts GL-Capstone-BBTM-G1

- artifacts
- TicketingPlatform.sol
- Interfaces.sol
- CustomerFactory.sol
- BookingFactory.sol
- AirlineFactory.sol
- FactoryRegistry.sol
- 1_Storage.sol
- 2_Owner.sol
- 3_Ballot.sol

scripts tests .deps README.md

Solidity code files are available in the Contracts folder

2. Approve Contract Deployment

factory.sol BookingFactory.sol AirlineFactory.sol FactoryRegistry.sol

MetaMask Notification - GL-Capstone-BBTM-G1

BBTM-ATM01 New Contract

New gas experience We've updated how gas fee estimation and customization works. Turn on Enhanced Gas Fee UI in Settings

https://remix.ethereum.org CONTRACT DEPLOYMENT

DETAILS DATA

Estimated gas fee 0.00323785 ETH
Site suggested Max fee: 0.00323785 ETH

Total 0.00323785 ETH
Amount + gas fee Max amount: 0.00323785 ETH

ENVIRONMENT Injected Web3 Custom (34343) network

ACCOUNT 0x8dc...903A9 (904625697)

GAS LIMIT 3000000

VALUE 0 Wei

CONTRACT CustomerFactory - contracts/GL-Capstone-BBTM-G1.sol

Deploy

Publish to IPFS

OR

At Address 0xd351486cA27eb5cBfbE73c

Transactions recorded 0

Deployed Contracts

TICKETINGPLATFORM AT 0XF13...73E

cancelBooking address _bookingId

makeBooking string _customerEmail, add

BBTM: In Action

The screenshot shows the MetaMask extension interface. On the left, there's a sidebar with various icons for managing Ethereum accounts. The main area is titled "DEPLOY & RUN TRANSACTIONS". Under "Deployed Contracts", it lists "TICKETINGPLATFORM AT 0XF13...73". Below this, a list of functions is shown with their parameters and transaction buttons:

- cancelBooking address _bookingId
- makeBooking string _customerEmail, add
- refundBooking address _bookingId
- registerAirline string _airlineName, string _airlineCode
- registerCustomer string _customerName, string _customerEmail
- registerFlight string _airlineCode, uint16 _flightNumber
- setFlightStatus address _flightId, uint8 _flightStatus
- getAllFlights string _airlineCode
- getAvailableSeats address _flightId
- getBookings string _customerEmail
- getFlights string _airlineCode, string _flightOrigin
- getFlightStatus address _flightId
- getRegistryAddress

At the bottom, there's a link to "Low level interactions".

Registering Eagle Airline & Flight between MAA & MUM

The screenshot shows the BBTM interface for interacting with the "TicketingPlatform". It has two main sections: "registerAirline" and "registerFlight".

registerAirline:

- Inputs:
 - _airlineName: "Eagle Airline"
 - _airlineCode: "EA001"
 - _airlineEmail: "eagle@sky.com"
- Buttons: "transact"

registerFlight:

- Inputs:
 - _airlineCode: "EA001"
 - _flightNumber: "1"
 - _flightOrigin: "MAA"
 - _flightDestination: "MUM"
 - _flightDepartureTime: "10000"
 - _flightArrivalTime: "20000"
 - _flightSeats: "10"
- Buttons: "transact"

Logs and Transactions:

- [block:249 txIndex:0] from: 0x8dc...903A9 to: TicketingPlatform.registerAirline(string,string,string) 0x021...d8de0 value: 0 wei data: 0x1a5...84251 logs: 0 hash: 0xecab...9aa2d transact to TicketingPlatform.registerAirline pending ...
- [block:250 txIndex:0] from: 0x8dc...903A9 to: TicketingPlatform.registerFlight(string,uint16,string,string,uint256,uint256) 0x021...d8de0 value: 0 wei data: 0xb41...00000 logs: 0 hash: 0x0ba...f52cb transact to TicketingPlatform.registerFlight pending ...
- [block:251 txIndex:0] from: 0x8dc...903A9 to: TicketingPlatform.registerCustomer(string,string) 0x021...d8de0 value: 0 wei data: 0x0ba...f52cb logs: 0 hash: 0xfb5...43249 transact to TicketingPlatform.registerCustomer pending ...
- [block:252 txIndex:0] from: 0x8dc...903A9 to: TicketingPlatform.registerCustomer(string,string) 0x021...d8de0 value: 0 wei data: 0x0ba...f52cb logs: 0 hash: 0x0ba...f52cb transact to TicketingPlatform.registerCustomer errored: MetaMask Tx Signature: User denied transaction ...
- [block:253 txIndex:0] from: 0x8dc...903A9 to: TicketingPlatform.registerCustomer(string,string) 0x021...d8de0 value: 0 wei data: 0x0ba...f52cb logs: 0 hash: 0x0ba...f52cb transact to TicketingPlatform.registerCustomer pending ...
- [block:254 txIndex:0] from: 0xa7386bBA929DA248F83577723AeE5dEE7B128AD7 to: TicketingPlatform.getFlights() data: 0x868...00000 call to TicketingPlatform.getFlights
- [call] from: 0xa7386bBA929DA248F83577723AeE5dEE7B128AD7 to: TicketingPlatform.getFlights(string,string,string) data: 0x868...00000 call to TicketingPlatform.getAvailableSeats

Watch videos in the **Artifacts → Demo Videos** folder for better understanding

BBTM: In Action

Registering a Customer & Searching a Flight

registerCustomer

_customerName: "Ankit"

_customerEmail: "ankit@bbtmgl1gl"

registerFlight string _airlineCode, uint16

setFlightStatus address _flightId, uint8 _fl

getAllFlights string _airlineCode

getAvailableS... address _flightId

o: string[]: 1B,2E,3E,4E,5E,6E,7E,8E,9E,10E

getBookings string _customerEmail

o: address[]: 0x37FD0C86B42B13620d2003E2
88a74FB892721d4A

getFlights

_airlineCode: EA001

_flightOrigin: MAA

_flightDestination: MUM

"accounts": {
 "account[0)": "0xa7386bBA929DA248F8357723AeE5dEE7B",
 "linkReferences": {},
 "transactions": [
 {
 "timestamp": 1650788175185,
 "record": {
 "value": "0",
 },
 },
],
},
"linkReferences": {},
"transactions": [
 {
 "block": 266, "txIndex": 0, "from": "0xa73...28AD7",
 "to": "TicketingPlatform.registerCustomer(string, string)",
 "data": "0xd9d...2d46",
 "hash": "0x0dd9...2d46",
 "call": {
 "from": "0xa7386bBA929DA248F8357723AeE5dEE7B",
 "to": "TicketingPlatform.getFlights(string, string, string)",
 "data": "0x0d08...b057f",
 },
 "transact": {
 "to": "TicketingPlatform.makeBooking(string, address, uint16)",
 "data": "0xc55...00000",
 "hash": "0x7d4...b2117",
 "call": {
 "from": "0xa7386bBA929DA248F8357723AeE5dEE7B",
 "to": "TicketingPlatform.getBookings(string, string)",
 "data": "0xa5d...00000",
 },
 "transact": {
 "to": "TicketingPlatform.cancelBooking(address)",
 "data": "0x823...21d4a",
 "hash": "0x78",
 },
 },
],
},
"listen on all transactions":

Seat Search, Booking & Cancellation

cancelBooking address _bookingId

makeBooking

_customerEmail: "ankit@bbtmgl1gl"

_flightId: "0x65DF5EE276d34d3C8DaE"

_seats: ["1B"]

refundBooking address _bookingId

registerAirline string _airlineName, string _airlineCode

registerCusto... string _customerName, str

registerFlight string _airlineCode, uint16

setFlightStatus address _flightId, uint8 _fl

getAllFlights string _airlineCode

getAvailableSeats

_flightId: "0x65DF5EE276d34d3C8DaE"

"accounts": {
 "account[0)": "0xa7386bBA929DA248F8357723AeE5dEE7B",
 "linkReferences": {},
 "transactions": [
 {
 "block": 266, "txIndex": 0, "from": "0xa73...28AD7",
 "to": "TicketingPlatform.registerCustomer(string, string)",
 "data": "0xd9d...2d46",
 "hash": "0x0dd9...2d46",
 "call": {
 "from": "0xa7386bBA929DA248F8357723AeE5dEE7B",
 "to": "TicketingPlatform.getFlights()",
 "data": "",
 },
 "transact": {
 "to": "TicketingPlatform.makeBooking(string, address, uint16)",
 "data": "0xc55...00000",
 "hash": "0x7d4...b2117",
 "call": {
 "from": "0xa7386bBA929DA248F8357723AeE5dEE7B",
 "to": "TicketingPlatform.getBookings()",
 "data": "",
 },
 "transact": {
 "to": "TicketingPlatform.cancelBooking(address)",
 "data": "0x823...21d4a",
 "hash": "0x78",
 },
 },
],
],
},
"listen on all transactions":

Watch videos in the **Artifacts → Demo Videos** folder for better understanding



Smart Contract Development - Challenges & Learnings

- 
1. Deciding on what functions to keep and what to remove when code size hits the limit of 24Kb. Solved by distributing the features across multiple smaller contracts by using **Interface Proxies** and **delegatecall** functions.



Responsibilities

01 Infrastructure - Configuring EC2 instances & Geth nodes - Ankit Shah & Kalaivanan S

02 Core Application - Base Contract Development - Preeti Aggarwal & Shuaib Mohammed

Artifacts

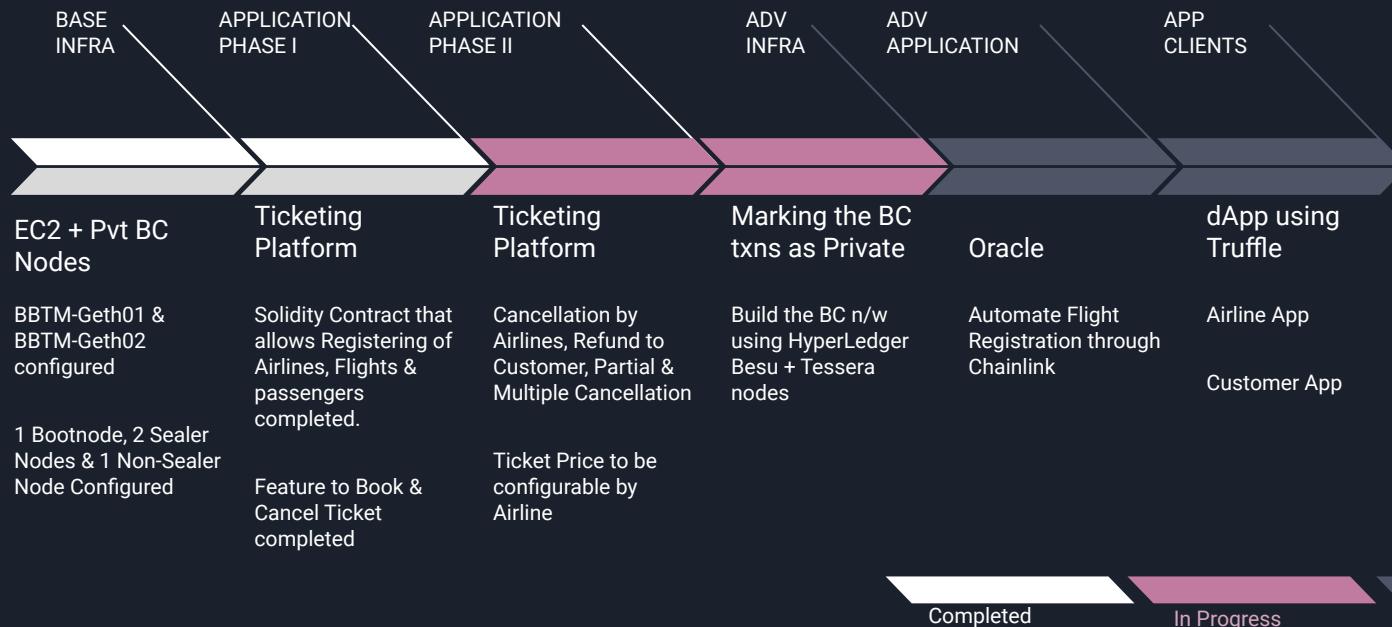
EC2 & BC Node Commands - Ankit Shah

Sequence Diagrams - Shuaib Mohammed

README - Preeti Aggarwal

Scope Document & Demo Screen Captures / Videos - Kalaivanan S

Project Status



Thank you!

