**Project Report**

| Name | Sujith T |
| --- | --- |
| Register No. | 22-IT-21 |
| Department | B.Sc Information Technology |
| Guide | Mr. K. Vijayakumar MCA., M. Phil., SET |
| Title | Secure TCP Reverse Tunneling via SSH on Cloud VPS for NAT/Firewall Traversal |

| Review No. | Date | Review Title | Signature | Remark |
| --- | --- | --- | --- | --- |
| 1. | 28-Aug-24 | Project Title, Organisation and tools |  |  |
| 2. | 12-Sep-24 | Abstract |  |  |
| 3. | 03-Oct-24 | System Study |  |  |
| 4. | 24-Oct-24 | System Design |  |  |
| 5. | 05-Dec-24 | Development Stage - I |  |  |
| 6. | 20-Jan-25 | Development Stage - II |  |  |
| 7. | 20-Feb-25 | Development Stage - III |  |  |
| 8. | 12-Mar-25 | Rough copy submission with Demo |  |  |
| 9. | 28-Mar-25 | Mock Viva voce |  |  |
| 10. | 04-Apr-25 | Fair Copy Submission |  |  |

**Title:**

Secure TCP Reverse Tunneling via SSH on Cloud VPS for NAT/Firewall Traversal

**Organisation:**

Sujith Thirumalaisamy,

CodeCrafters,

No. 6/405, Vellappanaicken Pudhur,

Manupatti (Po.),

Udumalpet (Tk.) - 642112,

Tiruppur (Dist.)

Ph: +91 7530097787

Email: hello@isujith.dev

| Frontend | React/Next.js |
| --- | --- |
| Design | ShadCN and TailwindCSS |
| Backend | Gin (Golang backend framework) |
| Database/ORM | PostgreSQL/Prisma |
| DevOps | Kubernetes deployment with Traefik as LB/reverse proxy |
| CI/CD | ArgoCD/ Github Actions |
| Version Control | Git/Github |

**Tools:**

**Abstract:**

The project **“Secure TCP Reverse Tunneling via SSH on Cloud VPS for NAT/Firewall”** Traversal aims to provide a secure solution for accessing devices or services located behind a NAT or firewall. By using SSH reverse tunnelling, the system allows users to expose their local services to the public internet via a cloud-based VPS, without directly opening ports on their local network. This is useful for accessing web servers, applications, or remote devices that are otherwise inaccessible due to network restrictions.

The project uses modern technologies such as React/Next.js for the frontend, Gin in Golang for the backend, PostgreSQL/Prisma for database management, and Kubernetes with Traefik for deployment. CI/CD is handled via ArgoCD and GitHub Actions, ensuring a streamlined and automated development process.

**Modules/Services:**

#### 1. User Interface Module (Frontend)

#### 2. Design System

#### 3. Authentication & Authorization

#### 4. Reverse Tunnel Management Service (Backend)

#### 5. Database Layer

#### 6. CI/CD Pipeline

#### 7. SSH Configuration Service

### **SYSTEM STUDY**

#### **1. Existing System**

Currently, accessing devices or services behind a NAT or firewall is challenging, with common solutions like port forwarding. Port forwarding involves manually configuring the router, which can be complex and exposes local networks to security risks such as DDoS attacks. Third-party services provide convenience but come with concerns around cost, privacy, and limited control over how traffic is handled. These existing methods either lack user-friendliness, pose security risks, or depend on external services that compromise privacy and scalability.

#### **1.1. Drawbacks of the Existing System**

* **Security Risks**: Exposed ports from port forwarding increase vulnerability to attacks like DDoS and brute force.
* **Complex Setup**: VPNs and port forwarding demand complex configurations, making them inaccessible to non-technical users.
* **Cost and Privacy Issues**: Third-party services incur subscription fees and potential privacy concerns due to traffic passing through their servers.

#### **2. Proposed System**

The proposed system addresses the limitations of existing solutions by offering a secure, self-hosted alternative using SSH reverse tunnelling. It enables users to expose local services behind NAT/firewalls without needing complex network configurations or third-party services. Leveraging SSH encryption and authentication, the system ensures secure data transmission while providing a user-friendly interface for easy tunnel management.

**2.1. Advantages of the Proposed System**

* **Enhanced Security**: SSH provides encryption, authentication, and secure tunnel management without publicly exposed ports.
* **Cost-Effective**: Open-source and self-hosted, reducing costs compared to subscription-based third-party services.
* **Automated CI/CD**: ArgoCD and GitHub Actions ensure seamless deployment, testing, and updates with minimal manual effort.

### **Modules/Services:**

1. **User Interface Module (Frontend)**

Built using Next.js, providing an interactive user interface for users to manage, authenticate, and configure services.

1. **Design System**

The design is implemented using ShadCN and TailwindCSS to ensure a responsive, clean, and efficient user experience across different devices.

1. **Authentication & Authorization**

Implements secure user authentication and authorization mechanisms using modern standards, ensuring only authorised users can create or manage tunnels.

1. **Reverse Tunnel Management Service (Backend)**

Built using the Gin (Golang) framework, this service handles the core logic of reverse tunnel creation and management via SSH, providing secure access to private resources.

1. **Database Layer**

PostgreSQL is used as the database, with Prisma ORM handling the data modelling and queries for user data, tunnel configurations.

1. **CI/CD Pipeline**

Managed through ArgoCD and GitHub Actions, ensuring automated testing, deployment, and delivery of updates to the system with zero downtime.

1. **SSH Configuration Service**

Handles the secure configuration and management of SSH tunnels, ensuring encryption and authentication processes are in place for safe reverse tunnelling.

### **DATABASE DESIGN**

**Table: User**

Primary Key: *id*

| **Field** | **Data Type** | **Width** | **Description** |
| --- | --- | --- | --- |
| id | VarChar | 36 | Primary key, UUID |
| name | VarChar | 100 | Name of the user |
| email | VarChar | 150 | Email address |
| emailVerified | DateTime | 25 | Timestamp of email verification |
| image | String | 255 | URL to profile image |
| createdAt | DateTime | 25 | Record creation timestamp |
| updatedAt | DateTime | 25 | Last update timestamp |

**Table: Account**

Primary Key: *providerAccountId*

| **Field** | **Data Type** | **Width** | **Description** |
| --- | --- | --- | --- |
| userId | String | 36 | Foreign key to User.id |
| provider | String | 50 | Authentication provider |
| providerAccountId | String | 100 | Unique account ID from the provider |
| refresh\_token | String | 500 | OAuth refresh token |
| access\_token | String | 500 | OAuth access token |
| expires\_at | Int | 11 | Token expiration timestamp (Unix) |
| token\_type | String | 20 | Token type (e.g., Bearer) |
| scope | String | 200 | OAuth scope |
| id\_token | String | 1000 | ID token |

**Table: Session**

Primary Key: *sessionToken*

| **Field Name** | **Data Type** | **Width** | **Description** |
| --- | --- | --- | --- |
| userId | String | 36 | Foreign key to User.id |
| sessionToken | String | 100 | Primary key, session token |
| expires | DateTime | 25 | Session expiration timestamp |
| createdAt | DateTime | 25 | Record creation timestamp |
| updatedAt | DateTime | 25 | Last update timestamp |

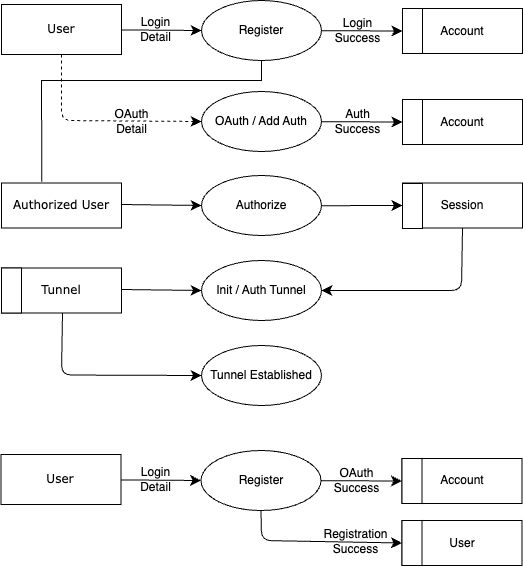
**Table: Tunnel**

Primary Key: *id*

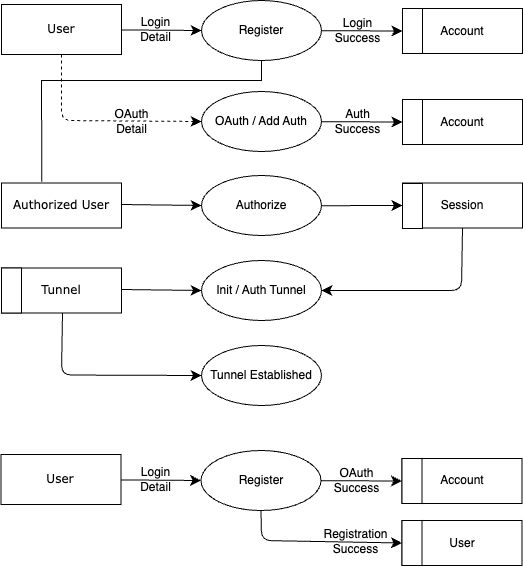
| **Field Name** | **Data Type** | **Width** | **Description** |
| --- | --- | --- | --- |
| id | String | 36 | Primary key, unique identifier |
| userId | String | 36 | Foreign key to User.id |
| domain\_slug | String | 100 | Unique domain identifier slug |

### **DATA FLOW DIAGRAM**

### **Level 0:**



### **Level 1:**



### 

### **ENTITY RELATIONSHIP DIAGRAM**

