

# MTE<sup>®</sup> API Relay for Secure Data Transmission

MTE API Relay is a server-to-server API gateway designed to protect sensitive HTTP traffic traversing uncontrolled or untrusted networks. Built on Eclipses' patented, quantum-resistant MicroToken Exchange (MTE) technology, it delivers encryption that exceeds conventional methods.

MTE API Relay is deployed in conjunction with TLS, not as a replacement—creating a layered, defense-in-depth model suitable for healthcare organizations requiring compliance, regulatory alignment, and quantum-safe future-proofing.

## BACKGROUND: What is an API and Why It Matters

Application Programming Interfaces (APIs) are the backbone of modern digital systems. They act as software intermediaries, allowing applications, databases, and services to communicate with each other securely and reliably.

- **In Healthcare:** APIs enable electronic health records (EHRs), lab systems, insurance platforms, and third-party applications to exchange critical patient data in real time. For example, a hospital EHR may call an API to retrieve lab results, verify insurance coverage, or share imaging data with a specialist system.
- **In General Use:** APIs power mobile banking apps, telehealth portals, logistics systems, AI/ML workflows, and IoT devices. Every digital service—from cloud apps to connected devices—relies on APIs to function.

### Utilization Trends

- APIs account for **over 80% of all internet traffic** in enterprise environments.
- In healthcare, API use is accelerating due to **FHIR mandates**, patient access requirements, and digital transformation initiatives.
- APIs are often **machine-to-machine** and unattended, making them a primary target for attackers who exploit stolen tokens, replay attacks, or TLS weaknesses.

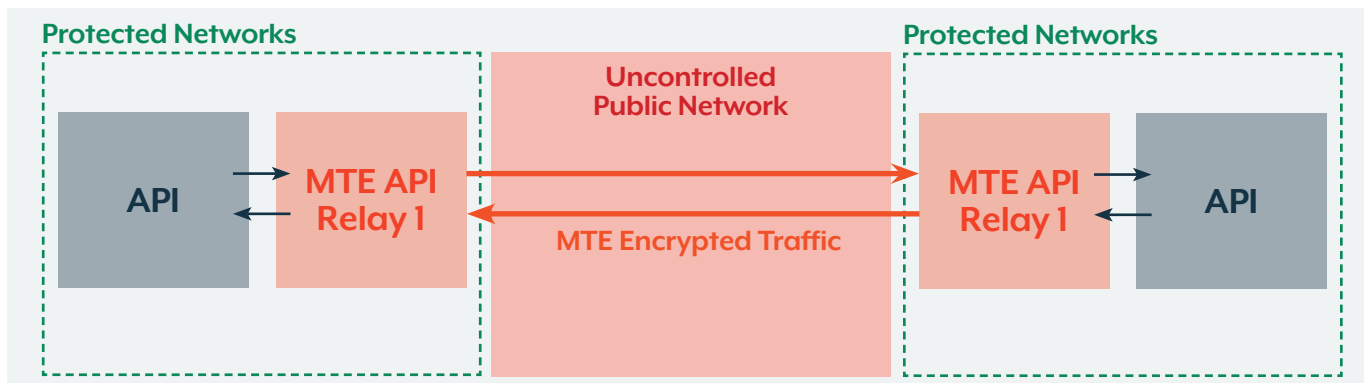
## WHAT IS MTE API RELAY?

### A Go-based, containerized application that:

- Encodes and encrypts HTTP requests at the sender.
- Transmits traffic securely across the network.
- Authenticates, decrypts, and proxies payloads to the target API.

This creates a **transparent, point-to-point secure channel** between two relay nodes—protecting sensitive data-in-motion without requiring application changes.

Unlike TLS-only approaches, **MTE provides independent payload-level protection validated under FIPS 140-3**, making it ideal for **healthcare APIs handling PHI and regulated workloads**.



# HOW IT WORKS

## 1. Deployment Model

- **Relay A:** Deployed alongside the client application or API service.
- **Relay B:** Deployed in front of the receiving service.

### Traffic Flow

1. Application sends HTTP request → Relay A.
2. Relay A validates headers, encrypts payload using MTE, forwards request.
3. Traffic transits over TLS (**double protection: TLS + MTE**).
4. Relay B validates, decrypts, and proxies to target API.
5. Response follows the same secure path in reverse.

## 2. Deployment Options

- **Azure Marketplace** – Delivered as a Kubernetes extension for AKS clusters. Supports cloud-native and hybrid healthcare workloads.
- [https://azuremarketplace.microsoft.com/en-us/marketplace/apps/eclypses1708534446098.eclypses\\_mte\\_api\\_relay?tab=Overview](https://azuremarketplace.microsoft.com/en-us/marketplace/apps/eclypses1708534446098.eclypses_mte_api_relay?tab=Overview)
- **On-Premises / Containerized** – Distributed as a Docker container, deployable on any container runtime or orchestrator. Enables secure bridging between **EHR systems, hybrid networks, and cloud APIs**.

## 3. Configuration

Relays are configured with environment variables:

- UPSTREAM – target API endpoint.
- CLIENT\_ID\_SECRET – shared secret for trust.
- OUTBOUND\_TOKEN – token for outbound requests.
- SECRET – key for inbound validation.

### Server Configuration

MTE API Relay is configured using environment variables.

### Required Variables

UPSTREAM – Upstream API or service URL.

CLIENT\_ID\_SECRET – Secret for signing client IDs (minimum 32 characters).

OUTBOUND\_TOKEN – Token appended to requests to denote the intended outbound recipient.

REDIS\_URL – Redis cluster for maintaining session pairs across load-balanced containers.

### Optional Variables

PORT – Default: 8080.

DEBUG – Set to true to enable verbose logs (default: false).

HEADERS – Object of custom headers.

CORS\_ORIGINS – Comma-separated list of allowed origins.

CORS\_METHODS – Default: GET, POST, PUT, DELETE.

### Full Example

```
UPSTREAM='https://api.my-company.com'  
CLIENT_ID_SECRET='2DkV4DDabehO8cifDktdF9eIKJL0CKrk'  
OUTBOUND_TOKEN='abcdefg1234567'  
REDIS_URL='redis://10.0.1.230:6379'  
PORT=3000  
DEBUG=true  
HEADERS='{“x-service-name”:”mte-api-relay”}'  
CORS_ORIGINS='https://www.my-company.com,https://  
dashboard.my-company.com'  
CORS_METHODS='GET,POST,DELETE'
```

## 4. Recommended Implementation

1. **Identify Integration Points** – APIs traversing untrusted/public networks (e.g., EHR cloud).
2. **Deploy Relays** – Relay A at source, Relay B at destination.
3. **Configure Security Parameters** – set variables above.
4. **Validate Connectivity** – test via /api/mte-echo.
5. **Monitor & Scale** – integrate with Azure Monitor / Grafana; add replicas as needed.

[https://www.youtube.com/watch?v=\\_UOwh\\_IJInM](https://www.youtube.com/watch?v=_UOwh_IJInM)

### Minimal Example

```
UPSTREAM='https://api.my-company.com'  
CLIENT_ID_SECRET='2DkV4DDabehO8cifDktdF9eIKJL0CKrk'  
OUTBOUND_TOKEN='abcdefg1234567'  
REDIS_URL='redis://10.0.1.230:6379'
```

## WHY HEALTHCARE NEEDS MTE API RELAY

- **Quantum-Resistant Encryption:** TLS alone is vulnerable to future quantum attacks. MTE is PQC-ready today.
- **Defense-in-Depth:** MTE protects the payload itself, ensuring data remains secure even if TLS sessions or certificates are compromised.
- **Compliance Alignment:** Strengthens adherence to HIPAA, PIPEDA, and PHI safeguards, supporting regulatory audits.
- **Zero-Trust Ready:** Protects machine-to-machine traffic across hybrid, multi-cloud, and cross-organization healthcare networks.
- **Securing API-to-API Communication:** Healthcare APIs increasingly handle real-time EHR, imaging, and clinical data flows. MTE adds a cryptographic shield beyond TLS.

