Security Built-in : Modulus Exchange

DNS Level

- Zone lockout for protected parts of the Exchange such as Admin Panel
- Anti-DDOS protection using HTTP Flood, UDP Flood, TCP Flood, Error based Detection, and QUIC Flood.
- Automatic HTTPS writes helps fix mixed content by changing “http” to “https” for all resources or links on your web site that can be served with HTTPS.
- Use of DNSSEC that protects against forged DNS answers. DNSSEC protected zones are cryptographically signed to ensure the DNS records received areidentical to the DNS records published by the domain owner.
- SSL/TLS certificates at origin to provide End to End Encryption
- HTTP Strict Transport Security (HSTS)
- Status: On, Serves HSTS headers with all HTTPS requests
    - Max-Age: 1 month, Specifies the duration HSTS headers are cached in browsers
    - Include subdomains: On, Every subdomain below this will inherit the same HSTS headers
    - Preload: On, Permit browsers to preload HSTS configuration automatically
    - No-Sniff Header: Send the “X-Content-Type-Options: nosniff” header to prevent Internet Explorer and Google Chrome from MIME-sniffing away from thedeclared Content-Type.
- Web Application Firewall’s Managed rules for common known CMS attacks.
- OWASP Hacker Proof, OWASP Core Ruleset (2013) provides protection against common attack categories, including SQL Injection and Cross-Site Scripting.These rules present a Challenge page when triggered.
- Bot Fight Mode, Challenge, and/or block requests matching patterns of known bots before they can access your site.

Code Level

- Request Rate Limits e.g. 10 requests per second per IP address.
- Self-hosted Server-side Captcha Validation
- Brute force attack prevention: We block the requesting IP for an hour as soon as the 10th invalid request is detected for an endpoint. e.g. HackerSpray
- Strong Password Policy, can be customized using a custom Regular Expression. Passwords are stored as Hash Strings in DB.
- CORS Enabled for specific domain only: Use of same domain CORS policy helps to restrict access from one domain to resources belonging to another domain.
- XSS Attack prevention:
    - Use of Regular Expressions to validate data & only store validated data.
    - XSS can be prevented by Encoding URL parameters using URLEncoder.

- Use of LINQ & ORM for SQL Injection attack prevention
- Use of HttpOnly cookies. Httponly flag is very important to avoid any XSS attack and has other benefits
- KYC profiles are encrypted with machine keys.
- Use of custom response headers (Appended below)

HTTP Response Headers

- x-content-type-options: nosniff
- x-frame-options: DENY
- x-xss-protection: 1; mode=block
- strict-transport-security: max-age=0; includeSubDomains; preload
- Content-Security-Policy=script-src 'self' object-src 'none'

JWT Bearer Token for User Session

- IP Specific
- AES Encrypted
- Single-use allowed with max 5 retries
- Disable all token issued prior to password change, after logout, and after every IP whitelist change
- Tokens are renewed every 5 minutes to mimic the server-side session timeout behavior.

Transactional OTPs or Email Verification Links

- Single-use tokens
- No retry allowed
- Google 2FA required after email verification
- Google 2FA required for address whitelisting

User profile & KYC Data

- Email OTP required for Email Change, OTP from both old and new emails.
- Email verification required for a Password reset
- KYC Profile, file upload sanitization
- File uploads to AWS S3 or Azure Storage using a pre-signed URL to prevent any foreign object from reaching the server where the object code executes.

Role-Based Authorization

- Custom role definitions to suit any situation
- Supports granular security permissions
- A full suite of role administration (list, get, create, update, delete)

Auditing

- Complete enterprise access and data auditing to meet compliance requirements
- HTTP Request / Response logging to track user activity (anonymous and authenticated users)
- Database change logging to track manipulation of data over time (anonymous and authenticated users)

Additional Security Protocols

Systems are compliant with Network Zoning policy and will employ the security team to help determine a compliant architecture.

Platforms deployed exclusively for internal use are located in an Internal-only backend, and under no circumstances located in DMZs or public frontends.

The secure filtering server/proxy holds no customer data and have no direct access to customer databases.

Production is not the same environment as stage, testing, development or pre-production. Each environment has a dedicated purpose .

Access to APIs from lower-security zones are provided over a secure communications channel.

The usage of encryption mechanisms is employed with enforcement of TLS 1.1 (or higher) using secure cipher suites and key length.

Access to the API is authenticated using distinct user accounts for each partner and, in case the API provides user level functions, distinct accounts for each user

In any API that differentiates between externally provided functions and others used only for internal purposes (such as administrative functions), the API separates both function sets. The separation enforces different communication paths such as separate web server instances and separate TCP ports.

The API implements a proper error handler. Stack traces and other internal information such as code snippets, filenames, or internal IP addresses are not be provided to the calling partner/service/process.

Any interfaces both for Internal users and customers are completely separated. Internal user's access is made only from corporate networks, never through public networks. For example, the Admin Panel is separate from customer UI. Networking restrictions are configurable via Cloud service provider or Cloudflare.

Any system that is modified or changed significantly is provided with updated documentation to the relevant Security Team contact.

Authentication to systems ios done using secure mechanisms especially for management purposes. Telnet use is not allowed and SSH must be used instead. Notes: clear-text authentication is not allowed.

Security patches are tested prior to deploying in a development environment. This avoids any potential problems when deploying them in the production environment.

Interfaces exposed outside the security domain provide authentication according to commercial best-practice standards.

Access to management interfaces and GUIs are done through bastion hosts/proxies deployed on management networks.

All data traffic coming from the Internet or other un-trusted networks terminates in a reverse proxy, which may validate and pass the request on to application servers. This reverse proxy physically separates trusted and un-trusted interfaces.

The network element is configured to automatically terminate administrative sessions after an configurable inactivity timeout period.

Data integrity is ensured by using referential integrity at database level.

System critical data is transmitted using resume functions like MD5 for ensuring data integrity during transmission.

All operations modifying system data (update, delete, insert,..) is made in transactions, ensuring there is no intermediate stage in the operation that can lead to data integrity problems. The entire operation is fully completed or fully incomplete.

All input-data processes (both manual-entry and automated-entry data) controls and validates the data entered in terms of formatting, length and correctness, for preventing system failures or attacks.

All data access, destruction and retention methods are carried out according to country specific legislation.

Documentation is prepared including, at least, the following:

(1) an explanation of the system architecture

(2) a list of platforms used (e.g. hardware, OS, middleware, applications and databases). Details about the version deployed are mandatory.

(3) details about interfaces with other systems/applications and protocols

(4) a detailed description of the network infrastructure according to the Network Security Zoning Standard

(5) an approved communication matrix and a topology map describing the connections.

Any account that is not personalized (e.g. 'root' or 'administrator') is detailed and flagged in the design document.

All new components are installed with the latest stable version with all security patches / service packs applied. The same patches are applied on already-running services in order to fix exploitable vulnerabilities.

Processes and systems are put in place to ensure that system continue to be patched, in a controlled fashion (consider testing, criticality of the system, likely impact of installing the patch), to remediate security vulnerabilities that may be discovered during the lifetime of the system.

All external interfaces of applications control syntactically and semantically the incoming requests before being processed by the application and have a deterministic behavior facing incoming requests.

The system provides measures to safeguard against user error and fraud (i.e. an application to check the validity of data entered or received from other applications).