QuillAudits

AUDIT REPORT

December 2025

For

EURASIA EXCHANGE

Table of Content

Executive Summary

Project Name	Eurasia Exchange
Protocol Type	Pentest
Project URL	https://uat.eurasiaexchanges.com/ https://uat.eurasiaexchanges.com/ - Admin Panel
Overview	Eurasia Exchange, also referred to as the Eurasia Digital Asset Exchange (EDX), is one of Cambodia's leading cryptocurrency trading platforms. It emphasizes a user-friendly experience with fast trade execution, strong security measures, and reliable performance- designed to make crypto trading accessible and smooth for all user types.
Review 1	3rd October 2025 - 11th November 2025
Updated Code Received	28th November 2025
Review 2	28th November 2025 - 2nd December 2025

Verify the Authenticity of Report on QuillAudits Leaderboard:

https://www.quillaudits.com/leaderboard

Number of Issues per Severity

Critical	0(0.0%)
High	4 (44.4%)
Medium	3 (33.3%)
Low	2 (22.2%)
Informational	0(0.0%)

Severity

Issues	Critical	High	Medium	Low	Informational
Open	0	0	0	0	0
Acknowledged	0	0	0	0	0
Partially Resolved	0	0	0	0	0
Resolved	0	**4**	**3**	**2**	0

Summary of Issues

Issue No.	Issue Title	Severity	Status
1	Broken Access Control IN ATTACHMENT ACCESS	High	Resolved
2	INSECURE DIRECT OBJECT REFERENCE IN P2P ORDERS	High	Resolved
3	INSECURE DIRECT OBJECT REFERENCE IN APPEALS SYSTEM	High	Resolved
4	EXCESSIVE ADMIN ACCESS TO USER PII AND ACTIVITY DATA	High	Resolved
5	PII DATA EXPOSURE IN PUBLIC MARKET ORDERS LISTING	Medium	Resolved
6	EXPOSED ENVIRONMENT CONFIGURATION FILE	Medium	Resolved
7	AWS S3 BUCKET MISCONFIGURATION	Medium	Resolved
8	CROSS-CONTEXT SESSION TERMINATION - FORCED LOGOUT DOS	Low	Resolved
9	UNRESTRICTED FILE UPLOAD WITH	Low	Resolved

Checked Vulnerabilities

- Improper Authentication
- Improper Resource Usage
- Improper Authorization
- Insecure File Uploads
- Insecure Direct Object References
- Client-Side Validation Issues
- Rate Limit
- Input Validation
- Injection Attacks
- Cross-Site Scripting (XSS)
- Cross-Site Request Forgery
- Security Misconfiguration
- Broken Access Controls
- Insecure Cryptographic Storage
- Insufficient Cryptography
- Insufficient Session Expiration
- Insufficient Transport Layer Protection
- Unvalidated Redirects and Forwards
- Information Leakage
- Broken Authentication and Session Management
- Denial of Service (DoS) Attacks
- Malware
- Third-Party Components

And More..

Techniques and Methods

Throughout the pentest of application, care was taken to ensure:

- Information gathering – Using OSINT tools information concerning the web architecture, information leakage, web service integration, and gathering other associated information related to web server & web services.
- Using Automated tools approach for Pentest like Nessus, Acunetix etc.
- Platform testing and configuration
- Error handling and data validation testing
- Encryption-related protection testing
- Client-side and business logic testing

Tools and Platforms used for Pentest:

Burp Suite	Acunetix	Nmap
DNSenum	Neucli	Metasploit
Dirbuster	Nabbu	Horusec
SQLMap	Turbo Intruder	Postman
Netcat	Nessus	And Many more..

Types of Severity

Every issue in this report has been assigned to a severity level. There are five levels of severity, and each of them has been explained below.

Critical: Immediate and Catastrophic Impact

Critical issues are the ones that an attacker could exploit with relative ease, potentially leading to an immediate and complete loss of user funds, a total takeover of the protocol's functionality, or other catastrophic failures. Critical vulnerabilities are non-negotiable; they absolutely must be fixed.

High (H): Significant Risk of Major Loss or Compromise

High-severity issues represent serious weaknesses that could result in significant financial losses for users, major malfunctions within the protocol, or substantial compromise of its intended operations. While exploiting these vulnerabilities might require specific conditions to be met or a moderate level of technical skill, the potential damage is considerable. These findings are critical and should be addressed and resolved thoroughly before the contract is put into the Mainnet.

Medium (M): Potential for Moderate Harm Under Specific Circumstances

Medium-severity bugs are loopholes in the protocol that could lead to moderate financial losses or partial disruptions of the protocol's intended behavior. However, exploiting these vulnerabilities typically requires more specific and less common conditions to occur, and the overall impact is generally lower compared to high or critical issues. While not as immediately threatening, it's still highly recommended to address these findings to enhance the contract's robustness and prevent potential problems down the line.

Low (L): Minor Imperfections with Limited Repercussions

Low-severity issues are essentially minor imperfections in the smart contract that have a limited impact on user funds or the core functionality of the protocol. Exploiting these would usually require very specific and unlikely scenarios and would yield minimal gain for an attacker. While these findings don't pose an immediate threat, addressing them when feasible can contribute to a more polished and well-maintained codebase.

Informational (I): Opportunities for Improvement, Not Immediate Risks

Informational findings aren't security vulnerabilities in the traditional sense. Instead, they highlight areas related to the clarity and efficiency of the code, gas optimization, the quality of documentation, or adherence to best development practices. These findings don't represent any immediate risk to the security or functionality of the contract but offer valuable insights for improving its overall quality and maintainability. Addressing these is optional but often beneficial for long-term health and clarity.

Types of Issues

Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

Resolved

These are the issues identified in the initial audit and have been successfully fixed.

Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

Partially Resolved

Considerable efforts have been invested to reduce the risk/impact of the security issue, but are not completely resolved.

Severity Matrix

Impact

Likelihood	High	Medium	Low
High	Critical	High	Medium
Medium	High	Medium	Low
Low	Medium	Low	Low

Impact

- **High** - leads to a significant material loss of assets in the protocol or significantly harms a group of users.
- **Medium** - only a small amount of funds can be lost (such as leakage of value) or a core functionality of the protocol is affected.
- **Low** - can lead to any kind of unexpected behavior with some of the protocol's functionalities that's not so critical.

Likelihood

- High - attack path is possible with reasonable assumptions that mimic on-chain conditions, and the cost of the attack is relatively low compared to the amount of funds that can be stolen or lost.
- Medium - only a conditionally incentivized attack vector, but still relatively likely.
- Low - has too many or too unlikely assumptions or requires a significant stake by the attacker with little or no incentive.

High Severity Issues

Broken Access Control IN ATTACHMENT ACCESS

Resolved

Description

The attachment download endpoint completely lacks authorization checks, allowing any authenticated user to access all files uploaded to the platform by simply modifying the attachment ID parameter. The system uses predictable sequential integer IDs starting from 1, making it trivial to enumerate through all attachments. Additionally, the P2P message endpoint accepts arbitrary attachment IDs in the attachment_ids[] parameter without validating ownership. During testing, changing the attachment ID from 20 to 19, 18, 17 successfully retrieved other users' private files, including bank statements, KYC documents, and personal chat attachments. The vulnerability exists both in direct S3 URLs and through the message API injection method.

Vulnerable CODE Location

https://uat.eurasia-s3-bucket-pub.s3.ap-southeast-1.amazonaws.com/uploads/attachment/uploads/{id}/file_*

/api/v2/p2p/message (POST with attachment_ids[] parameter)

Impact

This vulnerability allows complete access to all platform attachments including sensitive bank documents, government IDs, proof of address, and private transaction evidence. Attackers can systematically download the entire attachment database by iterating through sequential IDs. In a P2P trading platform handling financial transactions, this exposes users to identity theft, financial fraud, and severe privacy violations. The breach could affect thousands of users' confidential documents.

Steps to Reproduce

1. Create a P2P buy/sell order and mark it as complete to enable chat functionality
2. Upload any file through the chat interface
3. Intercept the POST request to /api/v2/p2p/message
4. Modify the attachment_ids[] parameter from your ID (e.g., 20) to any other number
5. The system accepts the modified ID and displays other users' files in your chat

Remediation

Implement strict authorization checks verifying the requesting user is either the sender or recipient of the attachment before serving the file. Replace sequential integer IDs with cryptographically secure UUIDs (version 4) to prevent enumeration attacks. Add server-side validation for the attachment_ids[] parameter in the message endpoint to ensure users can only reference their own uploads. Implement rate limiting on attachment endpoints to detect and block enumeration attempts. Consider adding file encryption at rest with user-specific keys.

POC

Request

Pretty Raw Hex

```
1  POST /api/v2/p2p/message HTTP/2
2  Host: uat.eurasiaexchanges.com
3  Cookie: _barong_session=7ed96ac3ffb349910acc7c248ffbdefd
4  Content-Length: 340
5  Sec-Ch-Ua-Platform: "macOS"
6  X-Csrf-Token: 1b7bb676210ac6b0b286
7  Accept-Language: en-GB,en;q=0.9
8  Sec-Ch-Ua: "Chromium";v="141", "Not?A_Brand";v="8"
9  Sec-Ch-Ua-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
   AppleWebKit/537.36 (KHTML, like Gecko) Chrome/141.0.0.0 Safari/537.36
11 Accept: application/json, text/plain, */*
12 Content-Type: multipart/form-data;
   boundary=----WebKitFormBoundarySXRtJI6UcqzXjJ1t
13 Origin: https://uat.eurasiaexchanges.com
14 Sec-Fetch-Site: same-origin
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Dest: empty
17 Referer: https://uat.eurasiaexchanges.com/p2p/paymentStep/26
18 Accept-Encoding: gzip, deflate, br
19 Priority: u=1, i
20
21 ------WebKitFormBoundarySXRtJI6UcqzXjJ1t
22 Content-Disposition: form-data; name="bid_id"
23
24 26
25 ------WebKitFormBoundarySXRtJI6UcqzXjJ1t
26 Content-Disposition: form-data; name="message[]"
27
28
29 ------WebKitFormBoundarySXRtJI6UcqzXjJ1t
30 Content-Disposition: form-data; name="attachment_ids[]"
31
32 17
33 ------WebKitFormBoundarySXRtJI6UcqzXjJ1t--
34
```

Response

Pretty Raw Hex Render

```
14 X-Amz-Cf-Pop: FCO50-P2
15 X-Amz-Cf-Id: nFE0vgVpDA4wTCKIawuxUQ51VTLSadcLduP99CLXO6wV_-d9RNo08g==
16 X-Xss-Protection: 1
17 X-Frame-Options: SAMEORIGIN
18 Referrer-Policy: strict-origin-when-cross-origin
19 Content-Security-Policy: img-src 'self' blob: uat.eurasiaexchanges.com
   data: https://uat-eurasia-s3-bucket-pub.s3.ap-southeast-1.amazonaws.com
   https://cdnjs.cloudflare.com;
20 X-Content-Type-Options: nosniff
21 Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
22 Access-Control-Allow-Credentials: true
23 Access-Control-Allow-Origin: https://uat.eurasiaexchanges.com
24 Vary: Origin
25 Access-Control-Expose-Headers: localhost
26
27 {
       "id":230,
       "state":"unread",
       "message":[
           ""
       ],
       "message_type":"bid",
       "bot_type":false,
       "created_at":"2025-10-28T10:33:53Z",
       "bid_id":26,
       "sender_id":"ID05C06BAE4A",
       "attachments":[
           {
               "attachment_id":17,
               "created_at":"2025-09-26T09:44:13Z",
               "uploads":
               "https://uat-eurasia-s3-bucket-pub.s3.ap-southeast-1.amaz
               onaws.com/uploads/attachment/uploads/17/file_175887985114
               7_ithr2.jpeg?X-Amz-Expires=600\u0026X-Amz-Date=20251028T1
               03353Z\u0026X-Amz-Algorithm=AWS4-HMAC-SHA256\u0026X-Amz-C
               redential=AKIAXQAMZUTIZB4P7R6Y%2F20251028%2Fap-southeast-
               1%2Fs3%2Faws4_request\u0026X-Amz-SignedHeaders=host\u0026
               X-Amz-Signature=60cef8bf72a07bc02457667c9a1fd485a791a9783
               65d17ff2c250db9048e6e68"
           }
       ]
   }
```

INSECURE DIRECT OBJECT REFERENCE IN P2P ORDERS

Resolved

Description

The P2P market orders API endpoint exposes complete order details through predictable sequential IDs without any authorization verification. Any authenticated user can access all platform orders by simply incrementing the order ID parameter from 1 onwards. The endpoint returns comprehensive trading information including full names, email addresses, payment methods, UPI IDs, transaction amounts, and order status. During testing, accessing /api/v2/p2p/market/orders/1 through /api/v2/p2p/market/orders/10 revealed complete order histories with sensitive financial data. The system makes no attempt to verify if the requesting user is involved in the transaction as either buyer or seller.

Vulnerable URL

https://uat.eurasiaexchanges.com/api/v2/p2p/market/orders/{id}

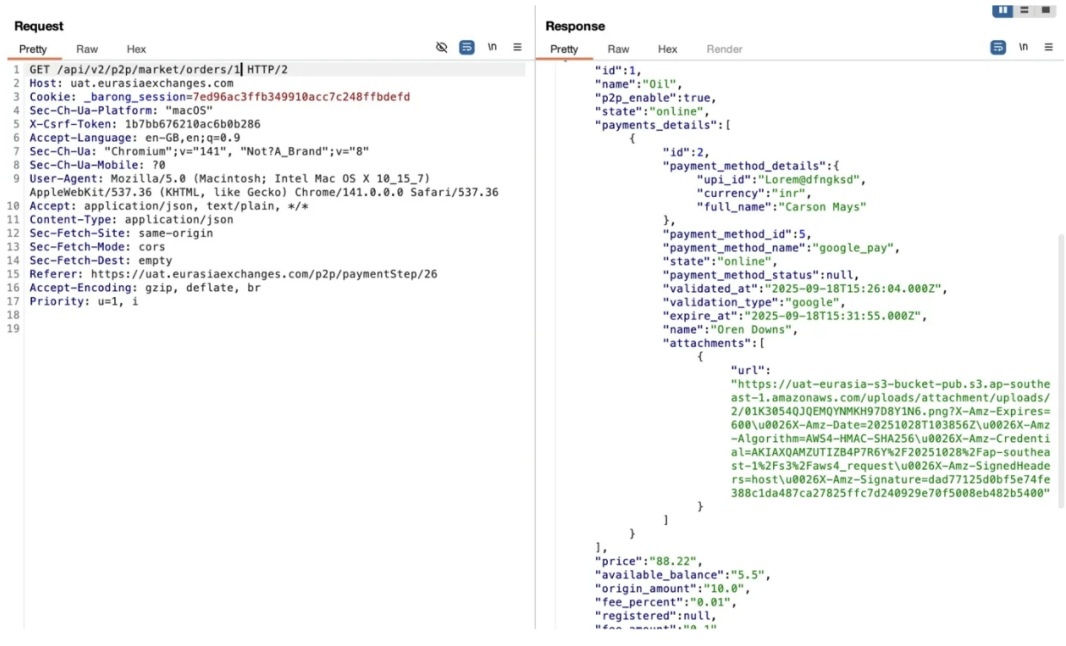
Impact

Attackers gain access to complete trading history, including payment processor details (Google Pay, UPI), exact transaction amounts, and user payment identifiers. This enables targeted phishing attacks using actual transaction data, competitive intelligence gathering on trading volumes, and building detailed profiles of high-value traders. The exposed UPI IDs and payment details could be used for social engineering attacks or unauthorized payment requests.

Remediation

Implement authorization middleware verifying the requesting user is either the buyer or seller before returning order details. Replace sequential IDs with UUIDs to prevent enumeration. For public order listings, return only essential trading information (price, limits) without any PII. Add field-level access control to mask sensitive data for non-participants. Implement audit logging for all order data access attempts.

POC



INSECURE DIRECT OBJECT REFERENCE IN APPEALS SYSTEM

Resolved

Description

The P2P appeals endpoint exposes complete dispute information through sequential ID enumeration without authorization checks. The system returns highly sensitive data including phone numbers, authentication tokens, dispute reasons, and evidence documents. Testing confirmed that changing the appeal ID from 5 to 4 exposed another user's complete appeal data including their phone number (+919216732344) and authentication token (ant8656f6af8e). The endpoint makes no verification whether the requesting user is involved in the dispute as appellant, respondent, or administrator. Appeals often contain the most sensitive information as users provide extensive evidence and personal details to resolve financial disputes.

Vulnerable CODE Location

https://uat.eurasiaexchanges.com/api/v2/p2p/appeal?id={id}

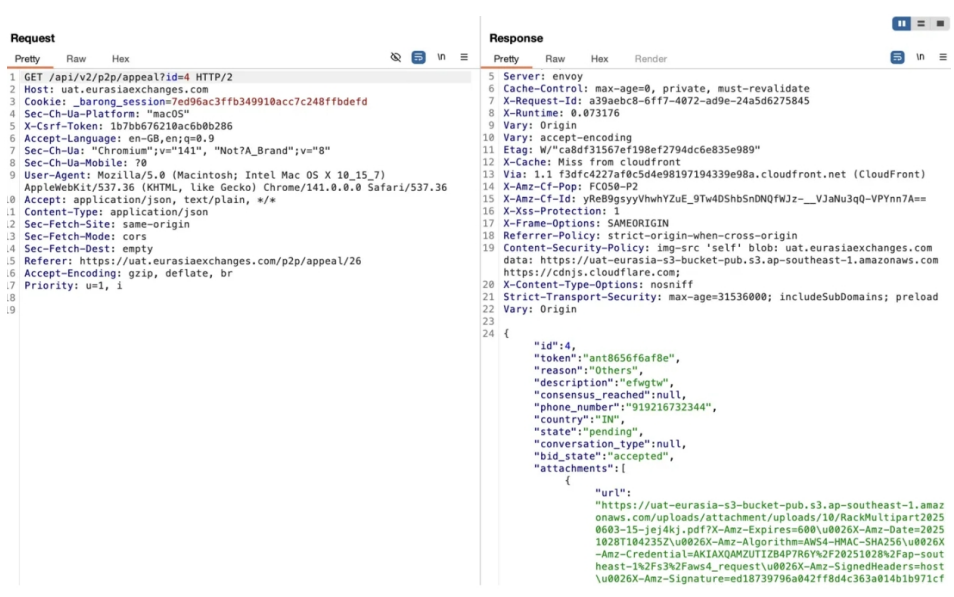
Impact

Direct exposure of phone numbers enables harassment, phishing via SMS, and social engineering attacks. The leaked authentication tokens could potentially be used for session hijacking or API access. Dispute information reveals financial problems, trading disputes, and personal conflicts that could be used for blackmail or targeted attacks. Access to evidence documents exposes additional sensitive materials submitted during dispute resolution.

Remediation

Implement role-based access control ensuring only involved parties (appellant, respondent, assigned admin) can view appeal details. Replace sequential IDs with UUIDs immediately. Add multi-layer authorization checking both authentication and specific role permissions. Implement comprehensive audit logging for all appeal access including timestamp, user ID, and accessed appeal ID. Consider encrypting sensitive fields like phone numbers at rest.

POC



EXCESSIVE ADMIN ACCESS TO USER PII AND ACTIVITY DATA

Resolved

Description

The admin activities endpoint exposes complete user activity logs with unmasked IP addresses, full email addresses, detailed timestamps, and browsing patterns without any data minimization or privacy controls. Admin users can access IP addresses in full (92.223.217.52), view all user emails (shivang.tester@gmail.com, test@gmail.com), track every login/logout event, monitor OTP requests, and see complete user agent strings. The system provides no audit trail of which admin accessed what user data, when, or for what purpose. This level of access violates privacy principles and enables potential insider threats where malicious administrators could stalk users, sell data, or conduct targeted attacks using the exposed information.

Vulnerable URL

https://uat.eurasiaexchanges.com/api/v2/barong/admin/activities?limit=10&page=1

Impact

Unmasked IP addresses enable administrators to track user physical locations and identify individuals behind accounts. The detailed activity logs allow building behavioral profiles for stalking or harassment. No audit trail means admin abuse cannot be detected or investigated. This violates GDPR data minimization principles and could result in significant regulatory penalties

Recommendation

Implement IP address masking showing only partial addresses (92.223.xxx.xxx) to prevent location tracking. Add role-based restrictions where different admin levels see different data granularity. Deploy a comprehensive audit logging recording of which admin accessed which user's data with timestamps. Implement data retention policies automatically purging old activity logs. Consider encrypting sensitive fields with admin-specific keys for accountability.

Medium Severity Issues

PII DATA EXPOSURE IN PUBLIC MARKET ORDERS LISTING

Resolved

Description

The public market orders listing endpoint returns extensive personally identifiable information without requiring authentication or authorization. The endpoint exposes email addresses in plaintext (krishnadeve12@gmail.com, naveen.beniwal@antiersolutions.com, mahimapreet@antiersolutions.com), exact account balances down to 15 decimal places (9.598231693493539), full names of traders, UPI IDs, and complete payment method details. This data is returned for all active orders in the system through a publicly accessible endpoint that should only display minimal trading information. The exposure occurs on the main trading interface where users browse available orders, making it accessible to anyone including unauthenticated visitors.

Vulnerable CODE Location

https://uat.eurasiaexchanges.com/api/v2/p2p/market/orders?order_type=bid&currency=usdt&state=online

Impact

Email addresses enable targeted phishing campaigns using actual platform context and trading data. Exact balance information allows attackers to identify high-value targets for focused attacks. Exposed payment details and UPI IDs can be used for payment fraud and unauthorized transaction requests. The data enables building detailed trader profiles for competitive intelligence or social engineering attacks.

Remediation

Remove all email addresses from public responses, using only member UIDs or masked versions (k***12@g***.com). Hide exact balances, showing only general availability indicators (Available/Limited/Unavailable). Implement data minimization returning only price, trading limits, and order ID for public listings. Move detailed information behind authentication and authorization checks. Consider implementing different API responses for public browsing versus authenticated trading.

POC

```
GET /api/v2/p2p/market/orders?
order_type=bid&currency=usdt&state=online&member_check=true&limit=10&page=1&ord
er_by=price&ordering=asc HTTP/2
Host: uat.eurasiaexchanges.com
Cookie: _barong_session=6a1def1d0ab2776fb623ebfa805f47a8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:144.0) Gecko/20100101
Firefox/144.0
Accept: application/json, text/plain, */*
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://uat.eurasiaexchanges.com/p2p/progress
X-Csrf-Token: 30260553b7d1b69c33eb
Content-Type: application/json
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: corsz
Sec-Fetch-Site: same-origin
Priority: u=0
Te: trailers
```

EXPOSED ENVIRONMENT CONFIGURATION FILE

Resolved

Description

A publicly accessible JavaScript configuration file exposes the complete internal API architecture, security settings, and system configuration. The file reveals all backend API endpoints (authUrl, tradeUrl, p2pUrl, rangerUrl), WebSocket connection details (wss://uat.eurasiaexchanges.com/api/v2/ranger), authentication flow information, captcha integration details including the site key (0x4AAAAAABbwIk4h_N-NTdT6), role hierarchy showing privilege escalation paths, and critically, shows development mode is enabled in production (devMode: true). The exposed configuration provides attackers with a complete map of the system architecture, timing configurations for security features (auto-logout: 60 minutes), and internal state machines. This significantly reduces the reconnaissance effort required for attacks.

Vulnerable CODE Location

https://uat.eurasiaexchanges.com/Wfqd5jBwyrmAxkQeGV3a7pKJ4tcDHFS9/env.js

Impact

Exposed API structure allows attackers to identify all available endpoints for testing and exploitation. Development mode active in production likely enables additional debug endpoints, verbose error messages, and reduced security controls. Role hierarchy exposure assists in privilege escalation attacks by revealing the authorization structure. Timing configurations help attackers optimize automated attacks to avoid detection.

Remediation

Move all sensitive configuration to server-side environment variables accessible only to the backend. Serve minimal client configuration dynamically based on user role and authentication status. Immediately disable development mode in production environments. Implement configuration encryption for any client-side settings. Use separate configuration files for different environments with proper access controls. Consider using a configuration management service with audit logging.

AWS S3 BUCKET MISCONFIGURATION

Resolved

Description

The AWS S3 bucket used for file storage is critically misconfigured with public read access enabled, completely bypassing the signed URL security mechanism. Files that should require time-limited signed URLs with authentication are directly accessible by removing all signature parameters. Testing confirmed that files accessible at signed URLs with X-Amz-Signature parameters remain accessible when all authentication parameters are stripped. The exposed URLs also leak the AWS Access Key ID (AKIAXQAMZUTIZB4P7R6Y) in plaintext. This means every file ever uploaded to the platform - including KYC documents, bank statements, chat attachments, and CSV exports - is publicly accessible to anyone with the direct URL.

Vulnerable CODE

Signed:

https://uat-eurasia-s3-bucket-pub.s3.ap-southeast-1.amazonaws.com/uploads/csv_storage/upload/11/-Order-1762802480.csv?X-Amz-Expires=600&X-Amz-Date=20251110T192120Z&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAXQAMZUTIZB4P7R6Y%2F20251110%2Fap-southeast-1%2Fs3%2Faws4_request&X-Amz-SignedHeaders=host&X-Amz-Signature=6a7ff8125e660b4f7f8377f5d6a74bbd6a45f5a92543529ef7cdd29415cb6c61

Public:

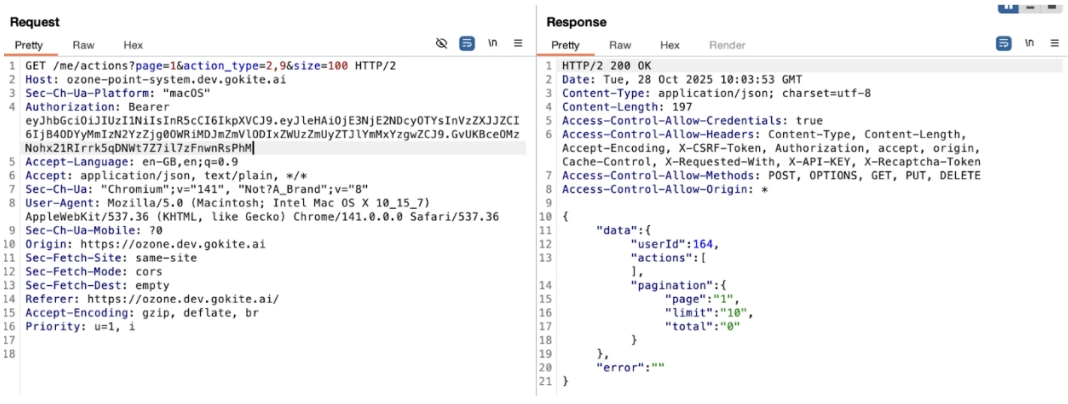
https://uat-eurasia-s3-bucket-pub.s3.ap-southeast-1.amazonaws.com/uploads/csv_storage/upload/11/-Order-1762802480.csv

Impact

Complete exposure of all platform files including sensitive KYC documents, financial records, and private communications. Signed URL expiration times are meaningless as files are permanently public. The exposed AWS access key could potentially be used for further AWS service exploitation. This represents a catastrophic data breach affecting every user who has ever uploaded a file to the platform.

Remediation

IMMEDIATELY block public access on the S3 bucket through AWS console bucket permissions. Implement bucket policy explicitly denying all public access and requiring valid signatures. Rotate the exposed AWS access key (AKIAXQAMZUTIZB4P7R6Y) immediately. Enable S3 access logging to identify any unauthorized access that has occurred. Implement CloudFront distribution with Origin Access Identity for secure content delivery. Conduct audit of all S3 buckets for similar misconfigurations.



Request

Pretty Raw Hex

```
GET /me/actions?page=1&action_type=2,9&size=100 HTTP/2
Host: ozone-point-system.dev.gokite.ai
Sec-Ch-Ua-Platform: "macOS"
Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3NjE2NDcyOTYsInVzZXJJZCI
6IjB4ODYyMmIzN2YzZjg0OWRiMDJmZmVlODIxZWUzZmUyZTJlYmMxYzgwZCJ9.GvUKBceOMz
Nohx21RIrrk5qDNWt7Z7il7zFnwnRsPhM
Accept-Language: en-GB,en;q=0.9
Accept: application/json, text/plain, */*
Sec-Ch-Ua: "Chromium";v="141", "Not?A_Brand";v="8"
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/141.0.0.0 Safari/537.36
Sec-Ch-Ua-Mobile: ?0
Origin: https://ozone.dev.gokite.ai
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://ozone.dev.gokite.ai/
Accept-Encoding: gzip, deflate, br
Priority: u=1, i
```

Response

Pretty Raw Hex Render

```
HTTP/2 200 OK
Date: Tue, 28 Oct 2025 10:03:53 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 197
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Content-Type, Content-Length,
Accept-Encoding, X-CSRF-Token, Authorization, accept, origin,
Cache-Control, X-Requested-With, X-API-KEY, X-Recaptcha-Token
Access-Control-Allow-Methods: POST, OPTIONS, GET, PUT, DELETE
Access-Control-Allow-Origin: *

{
    "data":{
        "userId":164,
        "actions":[
        ],
        "pagination":{
            "page":"1",
            "limit":"10",
            "total":"0"
        }
    },
    "error":""
}
```

Low Severity Issues

CROSS-CONTEXT SESSION TERMINATION - FORCED LOGOUT DOS

Resolved

Description

The platform suffers from a session management vulnerability where accessing the admin panel URL as a non-admin user triggers immediate session termination across all contexts. When a regular user navigates to or is tricked into visiting the admin URL (/Wfqd5jBwyrmAxkQeGV3a7pKJ4tcDHFS9), instead of simply denying access, the system terminates their entire session, logging them out from the main platform. This can be weaponized as a denial-of-service attack where attackers repeatedly force victims to visit the admin URL through various social engineering techniques. The vulnerability indicates improper session isolation between different application contexts and can be exploited to disrupt legitimate trading activities.

Attack Vectors

1. Phishing emails with admin URL disguised as “Verify your account”
2. Hidden iframes on malicious sites loading the admin URL
3. Shortened URLs (bit.ly) hiding the admin path
4. Social media posts with misleading links
5. QR codes resolving to admin URL

Impact

Targeted denial-of-service attacks preventing users from completing trades or withdrawals. Business disruption during critical trading periods causing financial losses. User frustration leading to platform abandonment and reputation damage. Can be weaponized for harassment by repeatedly forcing specific users offline during important transactions.

Remediation

Implement proper session context isolation where admin panel access attempts don’t affect main platform sessions. Return a simple 403 Forbidden error for non-admin users without session termination. Deploy admin panel on a separate subdomain (admin.eurasiaexchanges.com) with independent session management. Add warning interstitial page before any session-affecting actions.

UNRESTRICTED FILE UPLOAD WITH

Resolved

Description

The file upload functionality in the P2P message system performs no content validation, accepting any payload regardless of the claimed MIME type. Testing confirmed that uploading a file claiming to be image/png but containing HTML/JavaScriptpayload `<img src/onerror=prompt(8)>` was accepted. The malicious content executes when accessed by other users, creating a stored XSS vulnerability. The system trusts the Content-Type header without verifying actual file contents against magic bytes. Combined with the IDOR vulnerability in attachments, attackers can inject malicious files into any conversation, spreading payloads across the platform.

Vulnerable URL

/api/v2/p2p/message (POST with uploads[] parameter)

Steps to Reproduce

- Intercept a legitimate file upload request
- Modify the file content to contain XSS payload
- Keep Content-Type as image/png
- Submit the request - malicious content is accepted and stored

Remediation

Implement magic byte verification to ensure file content matches the claimed MIME type. Deploy content sanitization removing all potentially executable code from uploads. Establish a strict file type whitelist allowing only images (JPG, PNG) and documents (PDF) with proper validation. Implement Content Security Policy headers with script-src 'self' to prevent inline script execution. Add antivirus scanning for all uploaded files before storage.

Closing Summary

In this report, we have considered the security of the Eurasia Exchange. We performed our audit according to the procedure described above.

Some issues of High, medium, low severity were found. The Eurasia Exchange team resolved all the issues mentioned.

Disclaimer

At QuillAudits, we have spent years helping projects strengthen their smart contract security. However, security is not a one-time event threats evolve, and so do attack vectors. Our audit provides a security assessment based on the best industry practices at the time of review, identifying known vulnerabilities in the received smart contract source code.

This report does not serve as a security guarantee, investment advice, or an endorsement of any platform. It reflects our findings based on the provided code at the time of analysis and may no longer be relevant after any modifications. The presence of an audit does not imply that the contract is free of vulnerabilities or fully secure.

While we have conducted a thorough review, security is an ongoing process. We strongly recommend multiple independent audits, continuous monitoring, and a public bug bounty program to enhance resilience against emerging threats.

Stay proactive. Stay secure.

About QuillAudits

QuillAudits is a leading name in Web3 security, offering top-notch solutions to safeguard projects across DeFi, GameFi, NFT gaming, and all blockchain layers. With seven years of expertise, we've secured over 1400 projects globally, averting over $3 billion in losses. Our specialists rigorously audit smart contracts and ensure DApp safety on major platforms like Ethereum, BSC, Arbitrum, Algorand, Tron, Polygon, Polkadot, Fantom, NEAR, Solana, and others, guaranteeing your project's security with cutting-edge practices.

7+ Years of Expertise	**1M+** Lines of Code Audited
50+ Chains Supported	**1400+** Projects Secured

Follow Our Journey

AUDIT REPORT

December 2025

For

EURASIA EXCHANGE

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com audits@quillaudits.com