TECHNICAL SECURITY ASSESSMENT

Clyra Capital Fraud Infrastructure

JavaScript Component Analysis (Enhanced)

Document Classification:	PUBLIC
Advisory ID:	ADV-2025-002-JS-ENHANCED
Report Identifier:	ADV-2025-JS-7K9M4N6P8Q2R
Analysis Date:	November 13, 2025
Analyst:	SecureLeaf Cybersecurity
Target Infrastructure:	clyracapital.vip (Active)
Related Domains:	clyracapital.com (Neutralized)

SecureLeaf Cybersecurity Division
Dispensight

EXECUTIVE SUMMARY

WARNING - PUBLIC ADVISORY: This report is released publicly to protect potential victims from the Clyra Capital cryptocurrency investment fraud. Please share freely with anyone who may be targeted by cryptocurrency investment scams.

This document provides a comprehensive technical analysis of the JavaScript infrastructure deployed by the Clyra Capital cryptocurrency investment fraud operation, enhanced with live console capture data and API endpoint discovery. The analysis covers four primary JavaScript components discovered on the alternate domain clyracapital.vip, registered November 13, 2025, immediately following the neutralization of their primary domain.

Key Findings:

Finding	Severity	Description
DOM-based XSS Vulnerability	CRITICAL	Unsanitized user input in SVG/MathML contexts
Advanced Gamification System	HIGH	Sophisticated psychological manipulation infrastructure
Persistent Engagement Hooks	HIGH	Multiple fallback mechanisms to maintain victim engagement
Backend API Exposure	HIGH	Complete API endpoint structure discovered in production code
Link Manipulation System	MEDIUM	Automated lottery link interception and replacement
Production Debug Logging	MEDIUM	Extensive console.log() statements leak operational details
Asset Management Negligence	LOW	Unmodified stock photo filenames expose rushed deployment

THREAT ASSESSMENT: This infrastructure demonstrates sophisticated social engineering combined with catastrophically amateur security practices. Threat actors show persistence (immediate domain re-registration) but lack even basic operational security discipline. Production deployment includes verbose debug logging, exposed API structure, and unchanged stock imagery filenames creating a comprehensive operational intelligence goldmine for defenders.

INFRASTRUCTURE OVERVIEW

Component Architecture

The Clyra Capital scam infrastructure consists of four primary JavaScript components working in concert to create a comprehensive victim engagement and manipulation system. Each component serves a specific role in the overall scam operation:

Component	File	Primary Function	Lines of Code
Main Application	scam-site.js	Vue.js 3.5.13 frontend framework	46,600+
Configuration	config.js	API endpoint configuration (obfuscated)	12
Voting System	pk-vote-integration.js	Gamification and engagement tracking	260
Lottery System	lucky-draw-links.js	Link manipulation and user tracking	140

OPERATIONAL SECURITY FAILURES

1. Production Debug Logging (CRITICAL OPSEC FAILURE)

Live console capture reveals the scammers deployed with **ALL debug logging enabled**, providing real-time operational intelligence to anyone with browser developer tools:

Captured Console Output:

```
12:02:58.096 PK Integration - Starting initialization... 12:02:58.121 LuckyDrawLinks - Smart init,
document state: loading 12:02:58.385 LuckyDrawLinks - Initializing system... 12:02:58.387
LuckyDrawLinks - Processing links... 12:02:58.387 LuckyDrawLinks - Found 1 lottery links
12:02:58.388 LuckyDrawLinks - Replacing link: Object { originalHref:
"https://www.clyracapital.vip/lottery", targetHref:
"https://www.clyracapital.vip/workSpace/LuckyDraw/" } 12:02:58.389 LuckyDrawLinks - Link replaced
successfully 12:02:58.437 PK Integration - Page loaded
```

Impact: This leaks exact system initialization sequence, link manipulation mechanics in real-time, route detection patterns, complete operational workflow, and error conditions with retry logic.

Route Polling Evidence:

```
12:02:59.443 PK Integration - Route Check: {...} 12:03:00.454 PK Integration - Route Check: {...}
12:03:01.439 PK Integration - Route Check: {...} 12:03:03.497 PK Integration - Route Check: {...}
```

The 1000ms interval polling is clearly visible, confirming the persistent monitoring mechanism.

2. Complete API Endpoint Exposure

The production codebase contains **hardcoded API endpoints** with zero obfuscation:

```
check_invite_code: `${bn}/public/api/prize/check_invite_code`, prizeList:
`${bn}/public/api/prize/list`, lottery: `${bn}/public/api/prize/lottery`, prizeRecord:
`${bn}/public/api/prize/prize_record`, winningRecord: `${bn}/public/api/prize/winning_record`
```

Backend Structure Revealed:

- Base path: `/public/api/prize/`
- `check_invite_code` - Invite validation system
- `list` - Prize catalog retrieval
- `lottery` - Lottery draw mechanism
- `prize_record` - User prize history
- `winning_record` - Winner verification system

Impact: Complete API structure mapped without penetration testing, endpoint functionality clearly labeled, no authentication obfuscation visible, and direct testing targets identified.

3. Unmodified Stock Photo Filenames

The site uses **completely unmodified** stock photography with descriptive original filenames:

```
/img/diverse-group-of-colleagues-around-table.webp /img/hiker-looking-at-mountain-range.webp
/img/wmus-home-page-soccer-hero.webp /img/man-shopping-for-suits.webp
/img/mature-art-gallery-manager-writing-notes-3210x2140-1.webp /img/children-on-seesaw-in-park.webp
```

OPSEC Analysis:

- Zero asset customization effort
- Copy-paste deployment mentality
- 'wmus-home-page-soccer-hero.webp' suggests stock photo library source
- Dimension preservation (3210x2140) indicates no post-processing
- No attempt to obscure asset origins

Psychological Implication: Threat actors invested in sophisticated Vue.js frontend development but couldn't spend 30 seconds renaming files. This suggests template-based operation using pre-built fraud infrastructure, multiple deployments using identical asset libraries, speed prioritized over operational security, and possible automated deployment pipeline.

COMPONENT ANALYSIS: Main Application (scam-site.js)

Overview

The primary application bundle is a 46,600+ line minified Vue.js 3.5.13 application with integrated image viewing capabilities and extensive DOM manipulation features.

Technical Stack

- **Framework:** Vue.js 3.5.13 (latest at time of deployment)
- **Build Tool:** Vite (confirmed via console output and asset hashing)
- **Image Viewer:** Viewer.js integration with custom directives
- **State Management:** Vue 3 Composition API with reactive refs
- **Routing:** Client-side routing with history API manipulation

Critical Vulnerability: DOM-based XSS

CVSS Score: 8.6 (HIGH)

Attack Vector: Network / Attack Complexity: Low

Impact: Complete compromise of client-side application security

The application contains a critical DOM-based Cross-Site Scripting (XSS) vulnerability in its SVG and MathML rendering logic:

```
// Vulnerable sanitization logic zm = xl ? (e) => xl.createHTML(e) : (e) => e // Passthrough when Trusted Types unavailable // Dangerous sink with unsanitized input wd.innerHTML = zm( i === 'svg' ? `${e}` : i === 'mathml' ? `${e}` : e )
```

Exploitation Vectors:

- **SVG Script Injection:** `<svg><script>alert('XSS')</script></svg>`
- **Event Handler Injection:** `<svg onload="malicious_code()">`
- **ForeignObject Context:** `<svg><foreignObject><body onload="payload()">`
- **Image Tag XSS:** `<img src=x onerror="steal_data()">`

Potential Impact: An attacker could deface the site, redirect victims to warning pages, exfiltrate scammer data, or completely disrupt operations.

COMPONENT ANALYSIS: PK Voting System

The PK (Player Knockout) voting system is the psychological engagement engine of the scam operation. This 260-line component implements sophisticated gamification mechanics designed to build sunk cost fallacy and maintain daily victim engagement.

Engagement Mechanics

```
// Multiple route detection patterns const isPkRoute = currentPath === '/pk' || currentPath === '/pk.html' || currentPath.endsWith('/pk') || currentPath.includes('/pk.html') || currentHash === '#/pk' || fullUrl.includes('/pk.html'); // Persistent monitoring - checks every second setInterval(checkRoute, 1000);
```

Psychological Manipulation Tactics

Tactic	Implementation	Psychological Effect
Persistent Monitoring	Route checks every 1000ms	Ensures engagement hook always active
Multiple Fallbacks	5 delayed checks: 1s, 2s, 3s, 5s	Guarantees system loads regardless of timing
Forced Initialization	forceInitVoteSystem() on failures	Maintains engagement despite errors
Daily Ritual Creation	Daily voting requirement	Builds habitual behavior patterns
Points Accumulation	Vote rewards tracked	Creates sunk cost investment

BEHAVIORAL ANALYSIS

Victim Engagement Flow

The JavaScript components work together to create a comprehensive victim engagement pipeline:

1. **Initial Hook:** WhatsApp link → Landing page with Vue.js SPA
2. **Route Detection:** pk-vote-integration.js detects /pk route every 1000ms
3. **System Initialization:** 5 fallback attempts to ensure voting system loads
4. **Engagement Loop:** Daily voting creates habitual behavior pattern
5. **Sunk Cost Building:** Points accumulate, psychological investment increases
6. **Lottery Teasing:** Lottery links generate excitement about 'winning'
7. **Link Interception:** lucky-draw-links.js redirects to controlled workspace
8. **Qualification:** High-engagement users identified as premium targets
9. **Token Push:** Lottery 'requires' AEA token purchase or point spend
10. **Exit Scam:** Once invested, victim funds extracted

Live Operational Evidence

The captured console logs demonstrate **active deployment** on November 13, 2025. Empty hash indicates victims receive direct links to engagement systems rather than relying on client-side routing discovery.

THREAT ACTOR PROFILE

Skill Assessment

Capability	Level	Evidence
Social Engineering	Advanced	Sophisticated gamification, psychological manipulation
Web Development	Intermediate	Vue.js framework, modern JS patterns
Security Practices	Novice	XSS vulnerabilities, ALL production debug logs
Operational Security	Abysmal	Chinese comments, stock photo filenames, API exposure
Infrastructure Management	Intermediate	Multiple domains, CDN usage, rapid re-deployment

Attribution Indicators

- **Language:** Chinese comments ('jiekou qingqiu dizhi', 'ceshi') in production code
- **Framework Choice:** Vue.js heavily favored in Chinese development community
- **Time Zone:** Domain registration and deployment timing suggests Asia/Pacific operation
- **Infrastructure:** Singapore registrar (Gname), Cloudflare CDN common in Asia
- **Methodology:** 'Pig butchering' ('sha zhu pan') scam pattern with Chinese origins
- **Testing Notes:** 'ceshi' (test) in comments suggests development-to-production migration

Operational Characteristics

The threat actors demonstrate persistence through immediate domain re-registration, professional front-end with investment in UI/UX, but amateur back-end security with zero OPSEC discipline. Debug logs, filenames, and API exposure all in production. Copy-paste development with framework boilerplate and minimal customization suggests template operation with identical asset filenames indicating multi-deployment template.

INDICATORS OF COMPROMISE

JavaScript Fingerprints

- **Vue.js Version:** Exact version 3.5.13 identifiable in minified code
- **Viewer.js Integration:** Image viewer library with custom directive 'v-viewer'
- **Global Variables:** `window.globalSetting`, `window.voteSystemInstance`, `window.luckyDrawLinksInstance`
- **Console Strings:** 'PK Integration -', 'LuckyDrawLinks -' prefixes in all debug logs
- **Route Patterns:** `/pk`, `/pk.html`, `/lottery`, `/workSpace/LuckyDraw/` paths
- **Custom Classes:** `LuckyDrawLinks`, `VoteSystem` classes exposed globally

Network Indicators

- **Script Loading:** `/vote-system.js`, `/vote-styles.css` dynamically loaded
- **Workspace Path:** `/workSpace/LuckyDraw/` lottery redirection endpoint
- **API Base Path:** `/public/api/prize/` with 5 known endpoints
- **Asset Hashing:** Vite pattern: `index-[hash].js` (e.g., `index-Bwe0qHXI.js`)

Behavioral Indicators

- **Route Polling:** setInterval() checks every 1000ms for /pk route presence
- **MutationObserver:** Continuous DOM monitoring for lottery link injection
- **Multiple Fallbacks:** Delayed initialization attempts at 1s, 2s, 3s, 5s intervals
- **Console Verbosity:** Continuous debug output on every system action

Asset Fingerprints

Unmodified stock photography filenames provide infrastructure signature for content filtering, identifying related scam infrastructure, reverse image search, and file size/dimension fingerprinting for automated detection.

RECOMMENDATIONS

For Law Enforcement

- **Evidence Preservation:** Archive complete JavaScript source code, console logs, and network traffic
- **Backend Identification:** Monitor network requests to identify actual API endpoints (now mapped)
- **User Tracking:** WeakSet and console logs may reveal victim interaction patterns
- **Attribution:** Chinese language comments support Asia-based threat actor hypothesis
- **Coordinated Takedown:** Target all identified domains simultaneously to prevent migration
- **Template Identification:** Use stock photo filenames to identify related infrastructure

For Security Researchers

- **XSS Exploitation:** DOM-based XSS can be used for site defacement or victim warning
- **Console Monitoring:** Debug logs provide real-time operational intelligence
- **API Testing:** Complete endpoint structure now available for security assessment
- **Behavioral Analysis:** Route polling and MutationObserver patterns fingerprintable
- **Infrastructure Mapping:** Monitor for additional domain registrations with similar patterns
- **Asset Tracking:** Monitor for identical stock photo filename patterns across domains

For Hosting Providers

- **Content Scanning:** Scan for 'PK Integration' and 'LuckyDrawLinks' strings in JavaScript
- **Behavioral Detection:** Alert on 1000ms interval setInterval() patterns in SPAs
- **Vue.js Version:** Flag exact Vue.js 3.5.13 combined with lottery/voting systems
- **Asset Fingerprinting:** Scan for exact stock photo filename patterns
- **Console Log Detection:** Flag production deployments with extensive console.log() statements

For Potential Victims

WARNING SIGNS: If you encounter a website with: (1) Daily voting or "PK" competition systems, (2) Points accumulation for daily tasks, (3) Lottery systems requiring participation, (4) Promises of cryptocurrency token allocation, (5) Professional appearance with unsolicited WhatsApp contact, or (6) Console shows "PK Integration" or "LuckyDrawLinks" debug messages - **DO NOT ENGAGE**. This matches the Clyra Capital fraud pattern.

How to Check: Open browser developer tools (F12) and look at the Console tab. If you see messages like "PK Integration - Route Check" or "LuckyDrawLinks - Initializing", you are on a scam site.

CONCLUSION

The Clyra Capital fraud infrastructure represents a fascinating case study in operational security failure. The threat actors have created sophisticated psychological manipulation systems with advanced gamification mechanics, yet simultaneously deployed with such catastrophically poor operational security that they've essentially provided defenders with a complete intelligence package.

Key Takeaways:

- Threat actors show persistence through immediate domain re-registration
- Social engineering sophistication far exceeds technical security capability
- Production debug logging provides extensive operational intelligence
- Complete API structure exposed without any obfuscation
- Stock photo filenames unchanged revealing template-based operation
- DOM-based XSS vulnerability creates significant disruption opportunity
- Chinese language indicators support Asia-Pacific threat actor attribution
- Multiple domains indicate resilience planning but also pattern recognition opportunity

Comprehensive Intelligence Leakage

The scammers have essentially **published their own technical documentation** through: 14 unique console.log statements revealing every system operation, complete API endpoint structure with descriptive naming, unmodified asset filenames creating infrastructure fingerprints, Chinese language comments supporting attribution analysis, live console capture confirming November 13, 2025 deployment, XSS vulnerability enabling direct operational disruption, and real-time link replacement logged with full before/after URLs.

Operational Assessment

This is not a sophisticated cybercrime operation — this is a **social engineering operation built by people who Googled "Vue.js tutorial"** and forgot to disable development mode before deployment. The combination of advanced psychological manipulation tactics, professional-looking Vue.js frontend, zero production hardening, verbose debug logging in live environment, exposed API structure, unchanged stock photo filenames, and Chinese test comments in production code suggests an operation focused on speed and volume rather than operational longevity.

RECOMMENDED IMMEDIATE ACTIONS

CRITICAL — Time-Sensitive Actions: 1. **Hosting Provider Notification:** Hostinger (current host) with evidence package; GoDaddy (registrar) with abuse complaint 2. **Additional Domain Monitoring:** Watch for new registrations with Vue.js 3.5.13 + viewer.js combination, stock photo filename patterns, console log signatures, and API endpoint structure 3. **Victim Warning System:** Leverage XSS vulnerability to inject warnings, monitor console logs to identify active victims, track lottery link click patterns 4. **Law Enforcement Coordination:** Canadian Anti-Fraud Centre (victim location), Singapore authorities (infrastructure location), international coordination for takedown 5. **Intelligence Sharing:** Share indicators with hosting providers, publish console log signatures, distribute stock photo filename patterns, alert cryptocurrency exchanges about AEA token scam

DOCUMENT METADATA

Primary Analyst: SecureLeaf Cybersecurity Division (Dispensight) **Analysis Duration:** 6 hours **Evidence Sources:** Static code analysis (4 JavaScript files), Live console capture (November 13, 2025), Network traffic analysis, Asset enumeration, API endpoint discovery **Intelligence Classification:** PUBLIC — Please share freely to protect potential victims **Recommended Distribution:** • Post on social media to warn potential victims • Share with cryptocurrency communities • Forward to friends/family who discuss crypto investments • Provide to financial institutions and fraud prevention teams • Submit to anti-scam databases and registries **Next Review:** Monitor for infrastructure changes following takedown attempts

Report Identifier:	ADV-2025-JS-7K9M4N6P8Q2R
Contact:	SecureLeaf Cybersecurity Division
Email:	security@dispensight.com
Web:	dispensight.com

END OF ENHANCED ASSESSMENT