White Paper: API Protection Against Automated Attacks Using Obfuscation

Executive Summary

APIs are the circulatory system of the internet, with over 80% of traffic API-related, according to Akamai. API-powered web and mobile applications are now the principal way consumers access key services, including banking, brokerage, retail, travel, streaming, and gaming. Consumer actions within these apps trigger API requests to backend servers to facilitate authentication and data interchange. In all cases there is an implicit reliance on secure API connections to keep PII and transaction data safe and account access private.

The problem is externally-facing APIs, which power web and mobile applications as well as third-party business-to-business connections, have intrinsic security that is very weak. Because they must be publicly exposed and accessible for consumers and third parties to establish connections, they also are open to hackers seeking to exploit this accessibility. The ability of threat actors to accurately replicate API request syntax, and also to convincingly emulate a bona fide source of an API request, makes it challenging for traditional API security measures to block fraudulent activity. To make matters worse, hacker activity is typically automated via bot networks, and able to operate at massive scale. Costly problems such as account takeovers and credit card fraud result, and the sheer volume of fraud sometimes overwhelms and cripples API infrastructure.

SecureCo has introduced a novel way to block automated attacks before they ever get to the API server, using obfuscation technologies. Our solution establishes a secure connection between the API request source and the server endpoint, making API servers unavailable and inaccessible to illegitimate third parties. In effect, we provide an additional layer of network authentication that supplements the existing API security stack. The flood of unauthorized bot traffic typically hitting API servers never finds its target, and without access, the amount of unscreened fraudulent activity is dramatically reduced.

The Security Challenges for Externally Facing APIs

The native security of APIs is quite limited. API endpoints require requests to have a specific format and cryptographic key to be recognized as valid and generate a response. Unfortunately, it's quite straightforward to forge API request syntax, and API keys can be stolen and reused. Once a hacker learns how to communicate with the endpoint, they can map out the API functionality and available data through a trial and error process.

Often threat actors target the login process, with the goal of penetrating user accounts to steal funds, charge fraudulent transactions or capture sensitive data. However, bot-driven automated attacks can target many different points in the product flow, such as new account creation or the checkout process. In each case, the threat actor is falsifying API requests to automate a desired fraudulent outcome, such as fake new users or using stolen credit card numbers to transact.

Access to an API authentication flow provides hackers the potential to gain one of the most coveted objectives: the account takeover. There are many forms of attacks on the login process, such as brute force attacks and dictionary attacks, which do not leverage previously stolen passwords. But perhaps the most pernicious and effective of these threats are credential stuffing attacks. In this form of exploit, hackers leverage the criminal marketplace of 24 billion username and password pairs available on the dark web, stolen in various hacks over the years. Using a trove of these stolen credentials, they automate login attempts at massive scale. Since consumers tend to reuse their credentials across sites, the fraudulent login attempts have as high as a 2% success rate in account takeovers.

Costs Related to API Attacks

With the broad spectrum of different types of API attacks and the variability of different industries and business models, there are countless ways these attacks drive costs at affected organizations. Below, we highlight costs associated with account takeovers, which are one of the costliest forms of attack. After logging in as a legitimate user, the attackers can change passwords, giving them sole control over the account. Some typical adverse effects arising from account takeovers include:

- Financial losses: Stored value in brokerage, banking, credit, or loyalty accounts can be withdrawn and stolen.
- Fraudulent transactions: With access to consumer accounts, threat actors can engage in purchases, returns, and transfers to extract account value.
- Data theft: Attackers use the account to download and steal sensitive data, giving them a way to make money by selling it on the Dark Web.
- Overwhelming network resources: high volume automated attacks create heavy load on API endpoints, potentially resulting in denial of service, but in most cases increasing operational costs.

- High cost of remediation: In the aftermath of a data breach, the organization faces financial expenses arising from system recovery, notifying impacted customers, and legal fees associated with customer lawsuits.
- Reputation damage: New reports of the data breach harm how the public views the company.
- Customer churn: Customers may lose trust in the organization, causing them to switch their business to a competitor.
- Regulatory action: If a regulatory investigation indicates that the data breach arose from ineffective cybersecurity, the organization may have violated a legislative or regulatory compliance requirement and face fines or other sanctions.

API Protection Methods and Their Limitations

The core of the API vulnerability problem is that external APIs are broadly discoverable and accessible to anyone. This is particularly true for APIs serving consumer applications such as mobile apps. In the current mode of operation, it is necessary to ensure that all legitimate users can access the service anywhere from any device. Without any gating of access to the API endpoint at the network level, practitioners allow all requests to land and reactively sort out the legitimate from the fraudulent. Since by some estimates 70% of the requests are fraudulent, this approach leads to costly resource load, with spikes that sometimes cripple systems. For the filtering of legitimate API requests, practitioners are reliant on a patchwork of security solutions that have costly gaps.

Content Delivery Networks (CDNs)

CDNs are principally used to increase API performance and reduce latency by leveraging a global network of edge nodes. But CDNs also contribute to security by absorbing the peak load from high volume automated attacks. In this way, they contribute to continuity of service. CDNs create a buffer between internet threats and the API infrastructure, but mostly rely on additional tools such as WAFs (see below) to filter out fraud.

Multi-Factor Authentication (MFA)

MFA is the process of requiring users to provide two or more of the following before granting access to resources: 1) Something known (password), 2) Something possessed (mobile device, token), and 3) Something they are (biometrics like face ID).

Dual factor authentication is the most common implementation of MFA. While creating a significantly more secure authentication process, there are several reasons why MFA is not universally used. First off, it creates a more frictional user experience. End-users often resist the added inconvenience of MFA, and help desks are burdened with frustrated customers. In the competitive world of retail financial services, many leading

players have opted out of requiring dual-factor authentication for fear that a bad user experience will lead to customer churn.

Secondly, legacy protocols and authentication systems often do not support MFA. These systems simply cannot implement MFA until significant upgrades are made, leaving a window of opportunity for hackers.

In addition, MFA is by no means foolproof. Hackers have developed several methods of "MFA Bypass," which use sophisticated techniques to defeat this added layer of security. Some examples of these methods include MFA bombing (social engineering), session hijacking (phishing) and token theft, all of which steal or otherwise eliminate the need for the second authentication factor.

Finally, MFA only defends against automated attacks on authentication. To the extent an application is suffering from, for example, fraudulent credit card transactions from automated carding or card cracking, MFA will not help.

Web Application Firewalls (WAF)

A WAF monitors and analyzes all HTTP communications to block risky communications that attempt access to an application from the internet. These systems are effective for a broad range of threats but also provide features that reduce the threat of credential stuffing. WAFs can provide rate limiting, blacklisted IP ranges, and HTTP request analysis that can be successful in blocking fraudulent traffic.

A key challenge is that WAFs are a blunt instrument against credential stuffing that is not wholly effective. WAF filtering capabilities are straightforward for hackers to navigate around. Administrators must configure a ruleset, or signature, that aligns with known threats for the WAF to block them effectively. However, adversaries continually change their patterns and behaviors, and WAF signatures rapidly become outdated. It is onerous for admins to manage the continuous changing of signatures, so they are often outdated and ineffective.

Device Fingerprinting

Application providers can also use enhanced methods to identify legitimate end users, and apply a higher level of security to requests from parties they do not recognize. One approach to this is device fingerprinting, which entails collecting user characteristics such as browser type, operating system, and installed plugins to identify a unique pattern. When requests come from unrecognized devices to accounts associated with a known fingerprint, it may be a bot attack. Unfortunately, "may" is the key word here. While device fingerprinting can be part of an overall strategy, it falls to other tools to evaluate the legitimacy of the request. Simply blocking all requests from new or unrecognized devices would lead to a lot of real users getting shut out of their accounts.

Bot Detection

Bot detection tools analyze technical and behavioral visitor data, like user agent, IP owner, geolocation crawling speed, and crawling frequency. These systems search for activity or technical markers that do not align with plausible human behavior or device patterns. When threat actors use bots to deploy credential-stuffing attacks, bot detection technologies alert the security team to anomalous activity and block requests.

Threat actors have found ways to defeat most or all of the bot detection tactics. When they can't beat the defenses outright, they often appear real enough to be ambiguous. Blocking the pool of ambiguous requests would likely block legitimate users as well. Hackers use proxying and rotating IP addresses, inserting irregularity and human-like behaviors into request activity, and spoofing or blocking device fingerprinting. Through this arsenal of tactics, increasingly supercharged with the power of AI, attackers ensure that at least a portion of fraudulent requests penetrate the defenses.

API Security Platforms

API Security Platforms often have the threat detection features of WAF and bot detection aggregated into a single solution, along with other utilities to manage and monitor API attack surface. They function in part as command and control consoles to manage a host of security features, but do not significantly expand the defensive capability set beyond WAF and bot detection. As a result, for the same reasons that the above-listed API security solutions fall short, so do API security platforms.

The SecureCo CONNECT Solution

SecureCo's novel approach to API protection blocks fraudulent API requests from connecting to your API instructure in the first place, providing a pre-emptive, proactive approach. As an additional layer of defense-in-depth, SecureCo CONNECT works with existing protective measures to make API endpoints inaccessible to threat actors, blocking the full spectrum of bot attacks and fraud.

Key features of the SecureCo CONNECT solution include:

- Establishes a virtual tunnel connecting client applications to the API endpoint, creating an additional layer of network authentication to gain access
- Proprietary routing protocol uses obfuscation to avoid API call interception and reverse engineering
- Patented Rendezvous connection methodology delivers API requests despite closed ingress ports on the server side, making APIs undiscoverable and inaccessible other than via SecureCo routing
- Client-side agent or mobile SDK is simple to integrate; security without user friction.

- Server-side implementation compatible with existing security systems, e.g., bot-detection and WAF.

By preemptively blocking fraudulent requests, SecureCo CONNECT confers the following benefits:

- Restricted access to APIs leads to less server load and far fewer attacks.
- Resistance to API call interception and reverse engineering reduces API abuse risk
- Reduced risk of account breach and fraud; fewer resources to defend against and mitigate attacks.
- For mobile app users, Improved end user experience, with less frictional end user security policies.

SecureCo CONNECT is built on groundbreaking technology, including our patented Rendezvous connection methodology. This proprietary approach allows connections to be established with no open ingress ports, as the endpoints establish circuits by reaching out to a random, pre-agreed midpoint in the SecureCo STRATUS cloud delivery platform. In addition, data sent via this method is de-attributed using multi-layer encryption and routed via multiple proxies to hide the source and destination. Without the necessity of discoverable inbound ports, firewalls can now be set to block scans that identify connected services. APIs that escape detection avoid downstream security incidents.

SecureCo's networking solutions are high performance, and do not add meaningful latency to data transmission. Data transit is accelerated via the terabit backplane of best-of-breed cloud providers. In some cases, data transit speeds are comparable in performance to that of the customer's ISP unburdened by additional security.

SecureCo Implementation

SecureCo CONNECT is an agent-based solution that requires a light integration on both ends of the connection. The CONNECT agent enables the data to be routed securely over the STRATUS network, which is fully managed by SecureCo. The STRATUS security profile and performance can be tuned to client requirements.

For the server side implementation, a lightweight software service that operates on virtual or physical servers is configured and deployed in the customer environment on the corresponding network low side (DMZ) prior to any load balancer request handling.

Client-side implementations can either be accomplished via a SecureCo app install (available on all major platforms) or, in a mobile context, as an SDK integration into your application. Our fully documented mobility SDK supports both native and React frameworks as well as C++.

SecureCo data delivery was natively built for high demand applications, such as live video streaming. Our STRATUS network platform accelerates data transit by building upon the terabit backplanes of best-of-breed cloud service providers. As a result, latency is low – typically lower than most commercial SD tunnel solutions. SecureCo optimizes network performance through automated testing and by pruning bottlenecks.

Following a simple installation, SecureCo's managed data delivery platform requires minimal customer overhead. We offer a high service level and uptime SLA, with 24x7x365 support via chat/phone/ticketing.

Where SecureCo Fits In the API Security Market

There are many dimensions of API security, writ large, as illustrated in Figure 1 below. All the elements are very important to defending APIs against the full range of vulnerabilities. API security platforms attempt to cover the entire gamut of these functions, with varying degrees of success. As discussed earlier, defense in the areas of "Transport Security" and "Threat Protection" have historically had gaps, permitting a costly amount of fraud and abuse to pass undetected. SecureCo focuses specifically on these trouble areas, shutting down fraudulent activity.

SecureCo is compatible with all elements of the traditional API security stack. Refer to Exhibit 1 to view a reference architecture including SecureCo CONNECT. Our system operates adjacently to other defensive systems and does not require meaningful cross-integration. We work compatibility with WAF and bot detection solutions, even enhancing their effectiveness by providing identifying information that supports the fingerprinting of legitimate devices. SecureCo is an excellent complement to API Security Platforms, providing a differentiated security approach that results in a far more effective total solution.

Figure 1.: The API Security Spectrum

Development & Testing	Discovery & Inventory	Monitoring & Analytics	Transport Security	Threat Protection	Data Validation	Identity & Access
Security by design in API contract and policy design	Visibility and classification of all APIs for governance and protection of attack surface	Visibility of transactions and admin activities for detection of attacks and insider threats	Encryption, MitM defense, DDoS Prevention	Defense against exploits, stolen credentials, bot access	Conformance with API policies	Dynamic access control based on context-based decisions in runtime

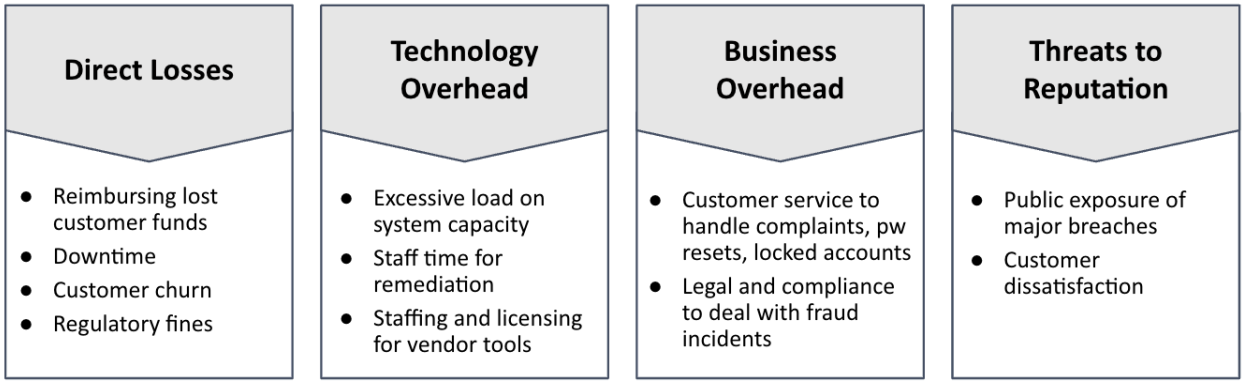
Refer to Exhibit 2 for a comparison of SecureCo's solution to WAF and bot detection, which are the most common API threat protection products in the market at present.

SecureCo ROI Considerations

Market analysts estimate the global costs of API insecurity to be in excess of $75 billion. Despite $3 billion in annual security investment by enterprises, API abuse is accelerating. Credential stuffing alone produces an average of 11 credential stuffing attacks every month per companies surveyed, and the annual average cost per affected company is $6MM (Source: FBI). And credential stuffing is only one of many types of automated attacks.

Many of the costs reflected in Figure 2 below could be averted with a greater defense-in-depth posture to automated API attacks. For the enterprise-scale customer, a SecureCo CONNECT implementation could result in a seven-figure savings across departments, resulting in project payback in only a few months, and an exceedingly high overall ROI.

Figure 2.: Cost Impact of Insufficient Security



Explore SecureCo's API Protection

SecureCo makes it easy to trial our API protection solution, with a low resource, easy Proof-of-Concept installation. Contact us today for a demonstration or more information: info@secureco.com.

Exhibit 1: API Protection Diagram: Mobile App Use Case

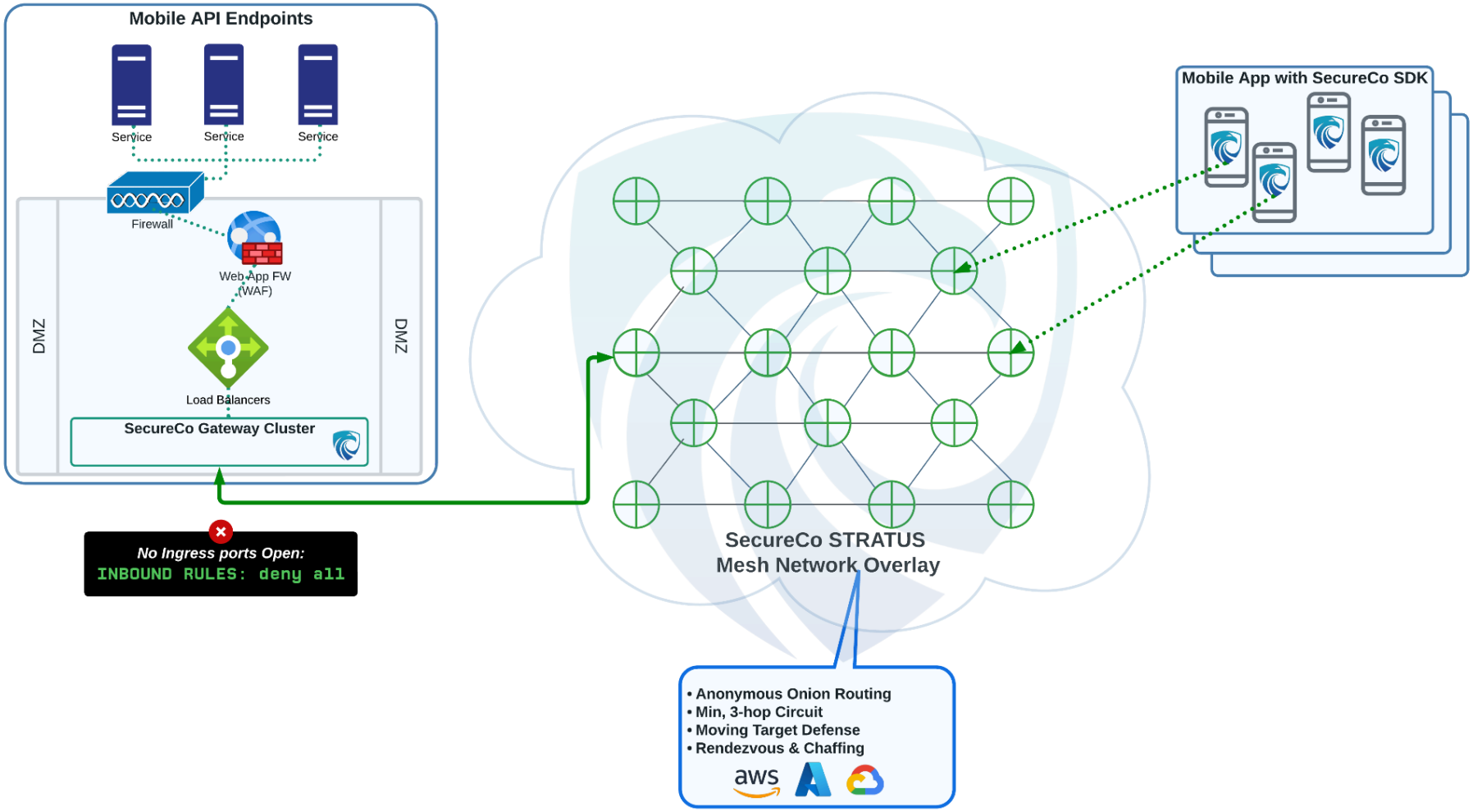


Exhibit 2: Comparison of API Protection Solutions

SecureCo solutions stand alone in denying API access to bots and attacker reconnaissance, preempting the perpetual cat-and-mouse game