# 

# 



Dynamic Site Accelerator (DSA)

Proof of Concept Test Results for [CUSTOMER\_NAME]

Akamai Account Team: [MAE]

[SE]

Date: [Date]

Table of Contents

Executive Summary 3

[CUSTOMER\_NAME] TEST Goals 3

Testing Methodology 3

Test Results 5

Recommendations 14

Appendix: 15

# Executive Summary

1

Using Akamai’s Dynamic Site Acceleration (DSA) service, we were able to demonstrate an [**X]% [location]** improvement in the delivery of a [test\_type]. The test was run from [**DATERANGE]** using Gomez testing services. The test mimicked a user **[workflow]**, and clearly shows a performance gain.

The graph below summarizes that performance gain. The horizontal axis is measurement points taken during the test, while the vertical axis is the duration of the entire transaction (in seconds). The yellow line indicates your server performance today, while the blue line is your performance improvement using Akamai.

**[Insert graph and one-line table of global performance comparing Akamai vs. origin ]**

Speed is a feature. It directly correlates with shopper conversion and abandonment rates.

# [CUSTOMER\_NAME] TEST Goals

* Determine the level of performance improvement that may be expected in the delivery of [CUSTOMER\_NAME]’s [origin\_url] on the Akamai platform.
* Provide a more consistent end-user experience, regardless of geographic location.
* Highlight potential problematic transaction steps.

# Testing Methodology

Two tests were run in parallel using the Gomez service. The first test was to baseline the current performance without using Akamai services. In this case, the test agents sent requests directly to [CUSTOMER\_NAME] data centre via [origin\_url].

The second test was intended to show the performance improvements using Akamai’s DSA services. Test agents were scripted to send requests to Akamai servers using a temporary trial property, [trialhostname].

The Dynamic Site Accelerator features that were enabled for this test included the following:

* **SureRoute for Failover** - Automatic failover to a static maintenance page if origin down
* **Edge Caching** – Temporary storage and reuse of content among many users
* **SureRoute for Performance** - Use of lower-latency paths through the Internet
* **Prefetching** - Retrieval of embedded page objects before actually requested
* **TCP Optimizations** – Reduction of “TCP Slow-start” and round-trip delays inherent in TCP
* **Edge Compression –** Compression of objects at the Edge server for Last Mile Acceleration
* **Persistent Connections** – Re-use of existing TCP connections
* **Tiered Distribution** – Hierarchical distribution of cached content

The type of testing used for this analysis was Gomez Backbone

**Backbone Monitoring** - measures Web page and transaction performance globally from major ISPs in real-time, enabling organizations to continuously monitor site availability and responsiveness.

The transaction used for testing consisted of the following individual steps:

[INSERT DETAILED TRANSACTION STEPS AS OBTAINED FROM CUSTOMER]

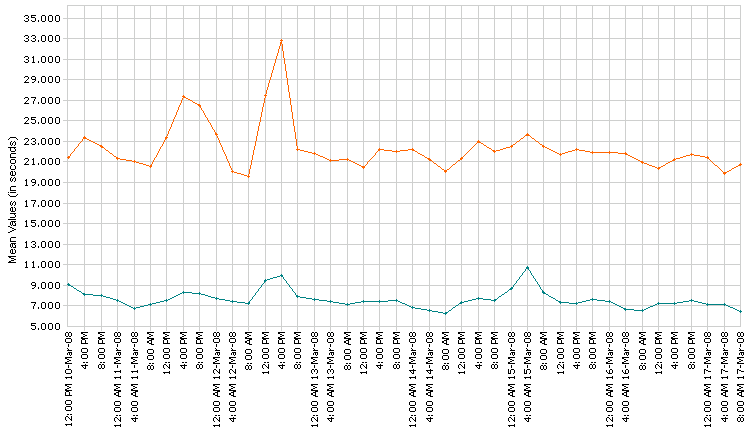
# 

# Test Results

The graphs below show consistent performance improvement during the test period as measured from all Gomez agents [**location**]. The graphs take the average performance across the agents used to measure [CUSTOMER\_NAME] site at hourly intervals.

Overall Transaction Performance [location]

The graph below depicts performance during the test duration as experienced on the origin versus the Akamai Intelligent Platform. Over the [X-day] period of the test, you can see Akamai (blue line) consistently outperforming origin (orange line).



 Akamai

 Origin

|  |  |  |  |
| --- | --- | --- | --- |
| **Performance** | **Origin (s)** | **Akamai (s)** | **Avg Improvement\* (%)** |
| Global |  |  |  |

\* Avg Improvement = (Origin-Akamai)/Akamai

Transaction Completion Analysis [location]

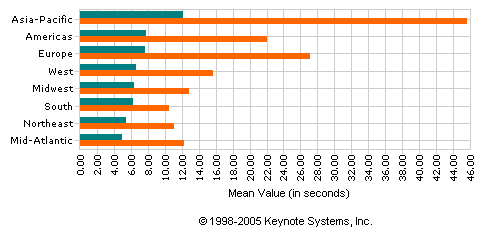
This graph compares transaction completion time between your origin server and Akamai. In this histogram, note the number of transaction steps Akamai is able to complete more quickly when compared to origin. Origin response times actually begin *after* Akamai has already begun completing transactions. Lastly, note the wider distribution of origin completion times when compared to Akamai.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **< 2s** | **< 4s** | **< 6s** | **< 8s** |
| Origin |  |  |  |  |
| Akamai |  |  |  |  |

Improvement in Performance by Region [location]

The graph below depicts performance by region of the world between the origin server and Akamai. Note that each region shows a performance improvement on the Akamai platform, especially as we distance from origin increases. (Note: North America is subdivided to see regional improvements across each geographical section - Northeast, South, and West).



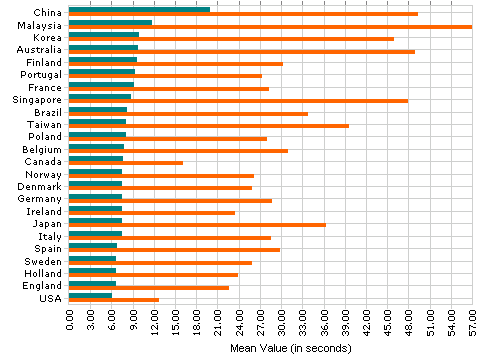
 Akamai

 Origin

|  |  |  |  |
| --- | --- | --- | --- |
| **Location** | **Origin (s)** | **Akamai (s)** | **Avg Improvement (%)** |
| Asia-Pacific |  |  |  |
| Americas |  |  |  |
| Europe |  |  |  |
| West |  |  |  |
| Midwest |  |  |  |
| South |  |  |  |
| Northeast |  |  |  |
| Mid-Atlantic |  |  |  |

Improvement in Performance by Country [location]

The graph below depicts performance by country between origin and Akamai. As seen in the previous graph, each country observes a improvement in performance with **[insert countries that the customer cares about]** seeing a **y**% improvement over the origin performance.



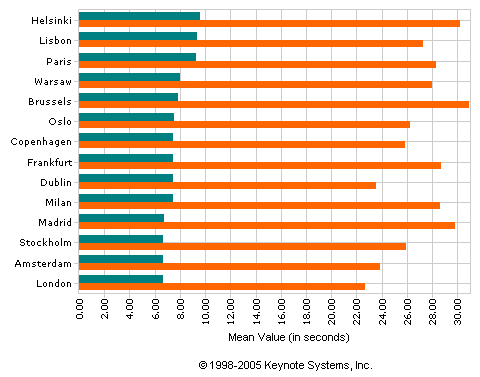
 Akamai

 Origin

|  |  |  |  |
| --- | --- | --- | --- |
| **Location** | **Origin (s)** | **Akamai (s)** | **Avg Improvement (%)** |
| China |  |  |  |
| Malaysia |  |  |  |
| Korea |  |  |  |
| Australia |  |  |  |
| Finland |  |  |  |
| Taiwan |  |  |  |
| Poland |  |  |  |
| Belgium |  |  |  |
| Canada |  |  |  |
| Norway |  |  |  |
| Denmark |  |  |  |
| Germany |  |  |  |
| Ireland |  |  |  |
| Japan |  |  |  |
| Italy |  |  |  |
| Spain |  |  |  |
| Sweden |  |  |  |
| Holland |  |  |  |
| England |  |  |  |
| USA |  |  |  |

Improvement in Performance by City [location]

[Add this graph if the customer requests region/country specific data] The graph below reveals the performance of origin versus Akamai as observed at the city level.



 Akamai

 Origin

|  |  |  |  |
| --- | --- | --- | --- |
| **Location** | **Origin (s)** | **Akamai (s)** | **Avg Improvement (%)** |
| Helsinki |  |  |  |
| Lisbon |  |  |  |
| Paris |  |  |  |
| Warsaw |  |  |  |
| Brussels |  |  |  |
| Oslo |  |  |  |
| Copenhagen |  |  |  |
| Frankfurt |  |  |  |
| Dublin |  |  |  |
| Milan |  |  |  |
| Madrid |  |  |  |
| Stockholm |  |  |  |
| Amsterdam |  |  |  |
| London |  |  |  |

Improvement in Transaction Steps - Comparative [location]

[Only add this section if the data is considered “good enough” to present to the customer]

The following graph details the comparative improvement in transaction steps for origin versus Akamai at for each step in the defined transaction. [Add any observed detail about each transaction step as appropriate]



|  |  |  |  |
| --- | --- | --- | --- |
| **Step** | **Origin (s)** | **Akamai (s)** | **Avg Improvement (%)** |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |
| 5 |  |  |  |
| 6 |  |  |  |

# Recommendations

NA

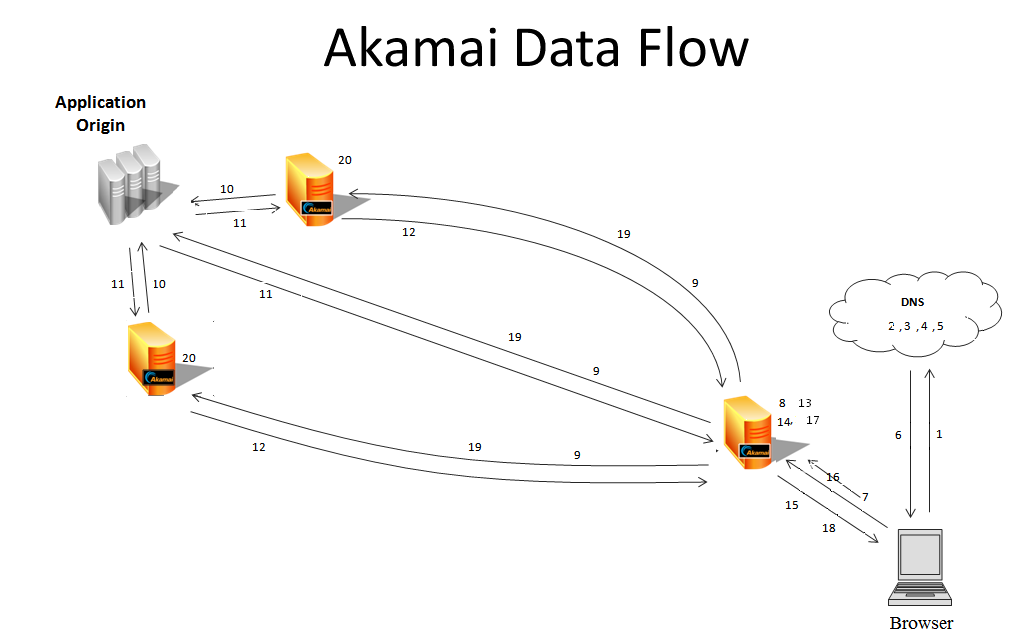
Testing Services are NOT the Real World

These test results represent the minimum performance improvement [CUSTOMER\_NAME] should expect to see in full production. Performance improvements typically increase when actual traffic load and real end user requests access our global network. Tests are subject to the following caveats:

* **[The bullets below should discuss performance features of the product that were not enabled for the test, but could lead to greater improvements for the customer.]**
* In a test environment, a 1-hour time-to-live (TTL) expiration on a website asset could produce a 0% cache hit rate. That same website, when subjected to real-world traffic, might see an 80-90% cache hit rate.
* Permanent connections are rarely in effect during this sort of interval test, and thus do not provide much benefit during the test period.
* 1-2 test agents per city were used for testing, while a diverse and dispersed set of users is considered more “real world.”
* Any existing origin DNS TTL settings can have a significant impact when only one request per city, per hour is being made.

# 

# Appendix:

**[Account team : Only include this appendix if the specific Site Delivery or Security information provided here is needed by the customer; otherwise delete this section]**

**Site Accelerator**

1) A user makes a request for www.customer.com. (During the POC, an Akamai-supplied test URL will be provided, such as www.customer.http.akamai-trials.com). The user’s browser makes a request its local name server for the IP address associated with the site.

2) Using the standard DSN resolution process, the local name server sends a request to the web site’s authoritative name server for the IP address associated with the web site.

3) The authoritative name serve replies with the CNAME www.customer.com.edgekey/edgesuite.net, which is required to allow the Akamai name server to determine which Akamai edge server to use. (During the POC a CNAME will not be required, as Akamai will supply a test URL.)

4) Using the standard DNS resolution process again, the local name server sends a request to the Akamai name server to determine which edge server to use.

5) The Akamai name server replies with the VIP (Virtual IP address) of the optimal Akamai edge server for the end user.

6) The local name server sends the IP address to the user’s browser.

7) The Browser request is sent to an optimal Akamai edge server.

8) The Akamai server uses the host header to locate a hostname-specific configuration in the metadata configuration file. The configuration file contains all the results for that specific hostname, including the Origin (origin.www.customer.com) DNS entry. The configuration also applies all caching and acceleration rules based on a request match. (For example, all .gif requests are cached for 1 day). Web Application Firewall (WAF) processing is also applied to the request.

9) The Akamai edge server uses SureRoute to determine the best path to pass the request to the customer’s data center. SureRoute periodically runs “races” to find the best path to the origin server. The best path can be direct (BGP) or through one of two alternate SureRoute “parent” servers located near the origin data center. SureRoute determines the best of these 3 possible paths.

10) The page request is forwarded to the Origin. This may or may not be proxied through a SureRoute parent server based on the route optimization described in Step 9.

11) The page response (Base Page) is then sent to the SureRoute parent server where TCP optimizations and compression are applied based on rules defined in the XML configuration file. If SureRoute determines that BGP is the fastest path to the origin server, then the response will go back over BGP to the Akamai edge server, where compression is applied.

12) The page response is then sent, compressed if applicable, to the Akamai edge server.

13) For the POC only, the base page is parsed, and embedded object tags are rewritten to allow those requests to be sent to the Akamai edge server rather than the Origin. For example, if a page contains a request to www.customer.com/images/logo.jpg, that string will be rewritten as www.customer.http.akamai-trials.com/images/logo.jpg. When the Browser subsequently parses this base page (Step 16), the request will then be sent to the Akamai edge server. This process is informally referred to as “Edge Akamaization”, and is only required for a POC. There is a small performance penalty for this process, therefore a production implementation of Akamai with a proper CNAME implementation will perform better than a POC using the Edge Akamaizer process. POC results should be considered conservative.

14) “Pre-fetching” is applied on the Akamai edge server. The server parses the page response looking for embedded object tags and makes asynchronous requests to the Origin. The Akamai edge server will first evaluate whether the embedded objects are in cache (if cacheable) before forwarding a request to the Origin.

15) Asynchronously with the prefetching step, the Akamai edge server then responds to the Browser with the page response.

16) The Browser parses the page (Base Page) looking for embedded object tags and makes synchronous requests, which are then mapped (per Step #5) to an Akamai edge server.

17) The Akamai edge server evaluates the cacheability of each object requested based on the rules in the hostname-specific XML configuration file.

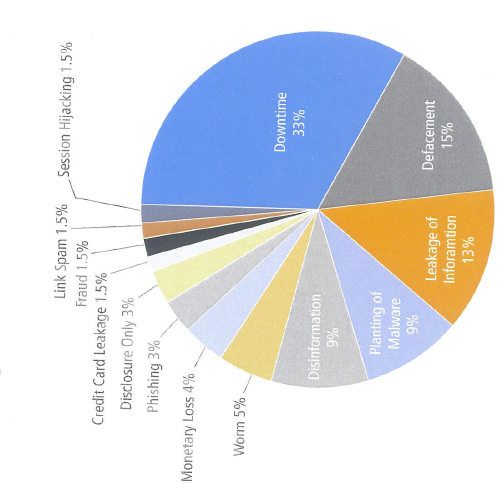
18) Objects that are in the edge server cache are returned to the browser, without sending a request to the Origin.

19) Objects that are missing from the cache, or expired, or objects that are designated as “no-store” are requested from the Origin using SureRoute.

20) The SureRoute parent servers are also edge servers, and they cache objects following the same configuration rules as the edge server near the end user. If the requested object is found in the cache of the SureRoute parent, it is returned to the Browser without sending a request to the Origin. This is referred to as a “midgress response”. Objects that are missing from the cache, or expired, or objects that are designated as “no-store” are requested directly from the Origin, and are then returned to the Browser (10, 11, 12, 15).

**Security**

Threat Environment- an evolving cast of “Chaotic Actors:”



Cyber crimes cost business more than $1Trillion each year. They can be driven by political motives, and may be linked to governments or organized crime. One executive at a large U.S. media company estimates that an attack, if not mitigated, could cost the business from $250,000 to $1 million per day

The most common attack is DDoS which accounts for 32% percentage of online threats. DDoS attacks are relatively cheap and easy to launch.

In a Forrester Research study, 74% of responding companies had experienced a DDoS attack within the past year. SQL injections are described as “the most pernicious vulnerability in human computer history” and form 21% of the threat. On average, a targeted web application can suffer 71 SOL injection attempts in an hour, and have scaled up to 1300 attempts per hour on specific occasions.

Site downtime, defacement of Web properties, and leakage of information together account for more than 60% of all the cyber attack outcomes, activities like phishing, monetary theft, planting malware for potential future efforts and committing fraud remain constant threats.

According to Gartner, two thirds of U.S banks planned to increase spending on fraud detection and online security systems in 2011, after a 6-year period that included more than 288 reported data breaches at financial services companies.

Akamai can amplify and multiply your security strategy.

The Akamai platform provides defence in depth moving the customer’s perimeter to the edge of the Internet. Many malicious attacks are mitigated by default. Additional services are available for fine-tuning the security perimeter.

**Akamai’s intelligent platform** carries up to 30% of the world’s HTTP traffic, and has an unmatched view into traffic patterns, geographic hotspots, and site-specific performance conditions across multiple industries.

**Platform Architecture** allows traffic from ports 80 and 443 only; all other traffic does not penetrate Akamai’s outermost security layer.

**Security solutions** including Kona Site Defender, Web Application firewall, Enhanced DNS, and Site Shied effectively spot, identify and mitigate threats before they reach the your infrastructure, ensuring continued application performance and availability, site integrity and compliance.

**Sheer** **size and scalability of Akamai’s Intelligent Platform** plays a key role in our ability to absorb and deflect malicious traffic, preserving the availability and performance of customers’ origin’ servers.

Akamai has protected its customers against everything from large scale DDoS attacks to cutting-edge Web application hacks. In July 2009, Akamai absorbed illegitimate traffic peaking at over 120 Gbps from more than 300,000 different IP addresses. As a result all Akamai customers withstood the attack without disruption, while non-Akamaized sites that were attacked suffered service disruptions, with many shut down for as long as two days.

During the week of Thanksgiving 2010, several Akamai customers saw thousands of times their normal request volume in highly distributed DDoS attacks as a part of a cyber-extortion attempt. By maintaining site availability, Akamai helped these customers avert $17M in lost revenue.

In late 2011, Anonymous targeted an Akamai customer with a sophisticated – but unsuccessful- cyber attack. Akamai provided a robust, multi-layered defence that automatically shielded against many types of illegitimate traffic, including SQL injection and other application layer attacks (Low Orbit Ion cannon and traffic from IP addresses making requests at too high a rate).

In Jan 2012, in response to the shutdown of file sharing site MegaUpload, DDoS attacks attributed to Anonymous targeted a number of websites belonging to government, media and Intellectual property enforcement organizations in the United States and Western Europe. Targeted sites that were delivered and protected by Akamai continued to operate normally during these attacks. Anonymous achieved several well-publicized successes at disabling targeted websites that were not on Akamai platform during these attacks.

Akamai security solutions deliver flexible intelligent cloud based defence capabilities that help organizations guard their perimeter and bolster security – without sacrificing performance.

Application Layer Security:

More and more cyber attacks are bypassing traditional firewall controls by using increasingly sophisticated HTTP-layer attacks to target Web sites and ap­plications. Unfortunately, the heterogeneous nature of Web applications, combined with continual, rapid development cycles, often leaves many doors open to exploit.

This trend drives the needs for firewalls and other security defenses that can understand and analyze Web traffic payloads such as HTTP, HTTPS, and XML — and provide protection against treacherous application-layer threats such as cross-site scripting (XSS), buffer over­flow exploits, and SQL injection attacks

Akamai delivers this type of protection at the edge of the network, augmenting traditional defense solutions with an unprecedented level of built-in redundancy and scalability.

**Web Application Firewall (WAF)**

Akamai’s Web Application Firewall service is a highly scalable defense system with the ability to detect potential attacks in HTTP and SSL traffic as it passes through the our platform, before it reaches your data center. WAF gives you the ability to set up traffic blocks or alerts based on rules that check for the presence of specific data like cookies, client certificates, and referrer fields, or detect anomalous and potentially malicious patterns in HTTP requests. Based on a translation of the open source ModSecurity core rule set (CRS), Akamai WAF’s protects against the most common and harmful types of attacks, including XSS and SQL injection.

WAF uses configurable, rule-based application layer controls to prevent the following types of generic attack vectors:

***SQL Injection*** : code injection technique that exploits a security vulnerability in a website's software

***Blind SQL Injection Attack*** : used when a web application is vulnerable to an SQL injection but the results of the injection are not visible to the attacker. The page with the vulnerability may not be one that displays data but will display differently depending on the results of a statement injected into the legitimate SQL statement used on the page.

***Cross-site Scripting (XSS) Attack*** : a type of injection problem, in which malicious scripts are injected into the otherwise benign and trusted web sites.

***Email Injection Attack***: a security vulnerability that can occur in Internet applications that are used to send email messages. A malicious user may exploit the MIME format to append additional information to the message being sent.

***HTTP Response Splitting Attack:*** occurs when data enters a web application through an untrusted source, most frequently an HTTP request, where it is included in an HTTP response header sent to a web user without being validated for malicious characters.

***LDAP Injection Attack:*** an attack used to exploit web-based applications that construct LDAP statements based on user input.

***Persistent Universal PDF XSS attack*** : affects the widely-used Adobe Acrobat Reader by abusing Acrobat’s open parameter features, opening a site up to cross-site scripting attacks if they host PDF documents.

***PHP Injection Attack*** : exploitation of a computer bug that is caused by processing invalid data. It can be used by an attacker to introduce (or "inject") code into a PHP page to change the course of execution

***Session Fixation:*** attempt to exploit the vulnerability of a system which allows one person to fixate (set) another person's session identifier (SID)

***SSI injection Attack:*** this vulnerability allows an attacker to inject code into HTML pages or even perform remote code execution.

***System Command Injection:*** inject and execute commands specified by the attacker in the vulnerable application

In addition to the above attacks, WAF has capability to protect against the following vulnerabilities.

**Bad Robots**:

Request Indicates a Security Scanner Scanned the Site

Request Indicates an automated program explored the site

Rogue web site crawler

**HTTP Policy**:

Request content type is not allowed by policy

Method is not allowed by policy

HTTP protocol version is not allowed by policy

URL file extension is restricted by policy

HTTP header is restricted by policy

**Outbound:**

ASP/JSP source code leakage

Cold Fusion Information Leakage

File or Directory Names Leakage

Directory Listing

Cold Fusion source code leakage

IIS Information Leakage

SQL Information Leakage

PHP Information Leakage

ISA server existence revealed

Microsoft Office document properties leakage

PHP source code leakage

Statistics Information Leakage

Zope Information Leakage

WebLogic information disclosure

The application is not available

**Protocol Anomalies**:

Request Missing a Host Header

Request Missing a User Agent Header

Request Missing an Accept Header

Host header is a numeric IP address

HTTP/0.9 Request Detected

Request Containing Content, but Missing Content-Type header

**Protocol Violations**:

URL Encoding Abuse Attack Attempt

Unicode Full/Half Width Abuse Attack Attempt

GET or HEAD requests with bodies

POST request must have a Content-Length header

Content-Length HTTP header is not numeric

Request Body Parsing Failed. %{REQBODY\_PROCESSOR\_ERROR\_MSG}

Invalid character in request

**Request Limits**:

Too many arguments in request

Network Layer Security

While cyber attacks are growing in sophistication, an increas­ing number of the most devastating attacks are focused on the application layer, the IP layer still accounts for nearly two-thirds of attacks today. Accordingly, defenses that harden this fun­damental layer of Internet communications are essential to the security of any Web application. Akamai leverages its unique architecture and real-time Internet knowledge base to offer a number of capabilities that help secure the network layer.

**SiteShield:**

Akamai’s SiteShield service helps protect your origin server by cloaking it from the public Internet — that is, removing it from the Internet-accessible IP address space. This mitigates risks associated with network-layer threats, including lower-level DDoS attacks that direct target the origin server.

SiteShield works by allowing the customer’s firewall to restrict incoming connections to Akamai SiteShield servers only, rather than leaving the standard HTTP/S ports 80 and 443 open and vulnerable to all incoming connections. Akamai SiteShield servers can be configured to communicate with the origin on non-standard ports as well to provide additional port masking protection. Aka­mai’s EdgePlatform intercepts and fulfills each end user request, communicating securely and “invisibly” with the origin server as necessary to retrieve content that is not in cache.

**Forrester Study:**

Based on a recently commissioned study "The Impact of Poor Web Site Performance in Financial Services" conducted by Forrester Consulting, 75% of online financial services consumers expect 99% or higher web site availability. The findings also indicate that website performance is second only to security in user expectations with 36% of online bankers - consumers who bank online - and 42% of online brokers - consumers who trade online - stating the importance of 100% availability.

Other highlights from the study:

64% of online banking and brokerage customers have had dissatisfying experiences online.

Web performance far and away is the biggest reason for the dissatisfaction, while site performance is second only to security in user expectations. Additional findings state that 56% of online bankers and brokers expect web pages to load in 2 seconds or less, which is significantly more than 47% of consumers who are just shopping online. Website performance ranks above even functions like single sign-on or a website that is easy-to-use.

Based on the feedback of 621 U.S. consumers who bank or trade online, Forrester Consulting reached the following key conclusions:

The web channel is vital to an ever-growing percentage of users. The importance of the Internet for financial services has been growing for years. 57% of online US adults bank online, and 36% of online US adults who have an investment account invest online. 29% of online bankers and 27% of online brokers access their accounts on a daily basis – a segment referred to as Power Users.

Website users have high expectations for website availability and page load speed. 75% of online bankers and brokers expect 99% website availability or higher. Additionally, 56% of online bankers and brokers expect web pages to load in two seconds or less.

Poor website performance leads to dissatisfaction more often then any other factor. 64% of online US bankers and brokers have had a dissatisfying experience when accessing their accounts. Web performance is far and away the biggest reason for this dissatisfaction with 54% of online bankers and 33% of online brokers expressing that sentiment.

The study also found that website performance has a direct impact on revenues, profits and satisfaction.  According to the study, three in ten web shoppers who encounter performance problems go to a competitor. In fact, 29% of users who encounter problems in the research process on a financial services website opt to go to a competitor’s website.  48% of online brokers who are conducting a transaction would use the phone or branch if they could not conduct their transaction online. The ultimate effect of poor performance is a decrease in willingness to recommend a firm with 48% of online bankers and brokers stating that poor performance “impacted” or “significantly impacted” their likeliness to recommend a firm’s services to a friend or family member. In addition, the study notes that the more engaged a user is online, the more profitable they are and the lower cost they are to serve.

3

3