



NETCONTINUUM

Attack and Intrusion Prevention

A Practical Approach to Reducing Risk

WHITE PAPER

The Web Security Problem

The ever-increasing desire of organizations to do e-Business has transformed the IT landscape. Web application development and deployment is accelerating rapidly as organizations open up applications and related business processes to a broader range of users in hopes of increasing revenue streams and reducing operational costs.

However, as organizations deploy new web applications under the pressures of a competitive landscape, they inadvertently open themselves up to a new breed of security vulnerabilities not addressed by traditional network security technologies.

Because of the pressures to deliver the new web application to market rapidly, security is rarely considered during the software development process. Developers create software with functionality in mind – not security. Web applications are designed to meet a business need, not to ensure the security of the system. Either the security policy is not generally available, or it just seems easier to postpone security concerns.

As a result, application security is most often addressed *after* the application has been deployed. As vulnerabilities are detected, developers scramble with patches and fixes in attempt to plug the holes, resulting in an endless and unmanageable number of security patches. The Computer Emergency Response Team (CERT) reported an average of 71 new vendor security patches released each week in 2002. These applications remain exposed to a wide range of vulnerabilities that hackers are targeting to gain illegal access and exploit the application.

The following statistical trends highlight the growing web security problem.

- **Application attacks are on the rise.** In an InformationWeek Research 2002 Global Information Security Survey fielded by PricewaterhouseCoopers, 30% of companies reported the hacking of known applications as a source of attack, up from 12% in 2001.
- **The number of security incidents is growing dramatically each year.** The Computer Emergency Response Team (CERT), an organization that tracks computer security incidents and vulnerabilities, reported 82,094 incidents in 2002, up from 52,658 in 2001.
- **The majority of attacks use port 80 and port 443.** Gartner Group recently reported that more than 70 percent of all attacks today target web and application layer vulnerabilities.

Myth: “We have a firewall and IDS and we are secure”

Businesses spend billions of dollars each year on firewalls, intrusion-detection systems (IDS) and other network security technologies in an effort to lock down their applications and systems from attack. And although these systems are a useful first line of defense against network-based attacks and intrusion, these defense mechanisms are useless in stopping web-based attacks. To pass web traffic, firewalls must

leave open TCP ports 80 and 443, providing absolutely no protection against attacks launched using HTTP and HTTPS.

IDS work by comparing network traffic against a known database of vulnerabilities and alerting administrators when intrusions occur. These systems are of little value for web security, as trying to detect application-specific attacks for every type of application is highly impractical. Many hackers are exploiting HTTP and HTTPS via Port 80 and 443, using web applications as a means of entry into the enterprise or to disrupt service availability.

No strategy or technology can prevent all potential attacks; a sophisticated hacker with adequate resources and a significant grudge can probably defeat many different defenses. However, by understanding the nature and scope of potential attacks, companies can create strategies for defending against a great majority of potential problems, balancing the cost and effort of security against risks to corporate data and service availability. To effectively prevent attacks we have to understand the scope of potential problems in the first place.

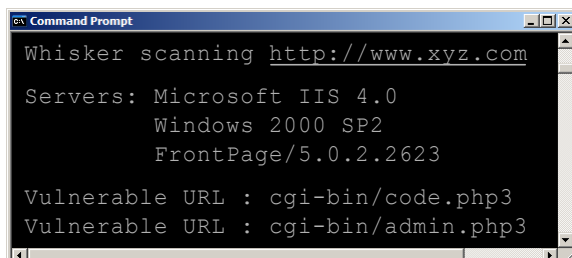
There are three basic steps hackers employ when launching attacks:

- Identify site information and vulnerabilities.
- Exploit known problems, both at the network and application layers.
- Cover any traces of the attack to prevent detection.

The Hacker's First Step – Scan for Vulnerabilities

The first step used by hackers when planning an attack is to assess the vulnerabilities of their target. Virtually all web sites give away their Server Type and Version, and most provide the server modules (SSL, PHP, PERL, etc.) and versions. Even the OS type and version on which the server is running are often delivered by default in the clear. These configurations arm hackers with a number of avenues to launch attacks against any site. Attackers can formulate a plan to exploit a site based on specifics of the products and topology detected.

A variety of automated tools, such as Whisker, Nessus, and Nikto, make preparing for an attack even easier. These tools probe the security of web sites, automating what once was a very tedious and manual process. This means that the sophistication and determination required of the hacker to scan a site for vulnerabilities is much lower than before. Web applications are now vulnerable to an ever-increasing universe of potential villains.



```
Command Prompt
Whisker scanning http://www.xyz.com
Servers: Microsoft IIS 4.0
        Windows 2000 SP2
        FrontPage/5.0.2.2623
Vulnerable URL : cgi-bin/code.php3
Vulnerable URL : cgi-bin/admin.php3
```

Figure 1: Automated programs scan sites for potential openings

Exploiting Web Applications

Once armed with information about the site, hackers may begin to probe for specific problems, try to discover unprotected resources, or even launch attacks against known applications. This phase requires no special tools other than a web browser; the URL is the carrier for most application attack payloads.

Discussed below are some of most common – and destructive – web application exploits, grouped generally by delivery method.

Input Validation

The single biggest cause of web application vulnerabilities is lack of proper input validation. Attackers can use these flaws to attack backside components, such as a database, through a web application. Hackers use input validation attacks to generate informational errors, to obtain arbitrary data access, or to execute commands, grab passwords, list directories, or copy files.

Many types of attacks fall into this category. Some of the more common are listed below:

- *SQL Injection Attacks* - Exploiting nonexistent or poorly designed input validation routines, SQL injection attacks represent a serious threat to any database-driven site. In a direct SQL Command Injection, an attacker creates or alters existing SQL commands to gain access to unintended data or even the ability to execute system-level commands on the host.
- *Direct OS or System Command Injection* - In some instances, a hacker can execute operating system commands by injecting them via HTML forms, cookies or a URL parameter. Using this type of attack the hacker is able to execute system-level functions such as removing and copying files, sending emails, and calling operating system tools to modify the application's input and output.
- *Meta-characters* - Some characters such as a semicolon, equals sign or a tilde have special meaning in scripts and applications. Hackers can change the behavior of a web-application by inserting meta-characters into URL-encoded parameters within query strings. Since many meta-characters are interpreted differently by different servers, the risk depends on the operating system, programming languages and workflow of the affected application. However, meta-characters can create huge security holes if allowed to penetrate the application.
- *Cross-Site Scripting* - This type of attack uses the web application as a mechanism to transport an attack to a client browser. A successful attack may disclose the user's session token, attack the local machine, or spoof content to fool the user. Cross-site scripting attacks cause users to execute malicious scripts unintentionally, often violating trust between the user and host web site which is, presumably, a trusted site. The security policy of the site may also be compromised.
- *Format String Attacks* - In a format string attack, an attacker changes the format specifications sent to a common program function like `printf()`, uncovering information about the system or even executing arbitrary code. Good programming practices can prevent the vulnerability, but the potentially problematic functions are widespread.

- *Path Traversal Attacks* - Path traversal vulnerabilities allow a hacker to execute commands or view data outside of the intended target path. Path traversal attacks are normally carried out via unchecked URL input parameters, cookies and HTTP request headers. Many web applications use the file system to save information temporarily and/or permanently. The file system may be accessed via numerous commands and functions used by the web application programming language. File system functions and commands can be very dangerous when carelessly used or publicly accessible. If a hacker knows the location of a sensitive file, he can retrieve the contents of the file by making an HTTP request containing the relative path

Forceful Browsing

Forceful browsing is very fundamental and easy to execute hacking technique that allows the attacker to jump directly to pages that can normally only be accessed through authentication mechanisms. By “guessing” the names of files and directories the hacker can view them without going through the business logic leading to those objects.

Buffer Overflows

Most application programs have fixed-size buffers that hold data in memory. A buffer overflow occurs when an attacker sends more data to the buffer than it was intended to hold. The extra data then “overflows” to adjacent buffers and can be executed as if it were a program.

Buffer overflows provide the hacker with a means to launch malicious code on the target server. That code may include commands to steal passwords or confidential information, alter system configurations, install backdoors, or launch other attacks

Parameter Tampering

Parameter tampering is a class of attack in which a hacker modifies data sent between the client and web application, such as URL query strings, form fields and cookies. Most web applications include a back-end database and the URL includes a SQL call to this database. Malicious users can manipulate the SQL code to potentially retrieve a listing of all users, passwords, credit card numbers, or any other data stored in the database.

Common attack types that fall under this category are cookie poisoning, HTTP header manipulation and form field manipulation.

- *Cookie Poisoning* - Cookie Poisoning is the modification of a cookie by a hacker to gain unauthorized information about a user, typically for the purposes of identity theft. The hacker will then use this information to gain access to the user’s accounts or fraudulently open new accounts.
- *HTTP Header Manipulation* - HTTP headers are control information passed between web clients and web servers over HTTP. Since HTTP request headers originate on the client, they may easily be altered by an attacker to include meta characters to take control of the system or manipulate cookies to steal another user’s login and access their account.

- *Hidden Form Fields* - Hidden HTML form fields typically hold system passwords or information such as merchandise prices in an e-commerce application. These “hidden” fields can be seen by performing a ‘view source’ on the web page. Some developers make the mistake of passing application configuration parameters to backend programs with hidden fields. Whether these form fields are pre-selected, free form or hidden, they can all be manipulated by a user to submit whatever values he/she chooses. For example, a user could use hidden form field to modify the price of an item for sale on an e-commerce site to purchase it at little or no cost. This can be accomplished by simply saving the page using view source, editing the HTML and re-loading the page in the web browser.

Disguising Attacks with Unicode and URL Encoding

Unicode was developed to allow a Universal Character Set (UCS) that encompasses most of the world's writing systems. Unicode Encoding is a method for storing characters with multiple bytes. Wherever input data is allowed, data can be entered using Unicode to disguise malicious code and permit a variety of attacks.

URL encoding allows non-alphanumeric characters in URL strings so that regular alphanumeric characters and symbols presented on most keyboards could be used. Certain web servers can be fooled by nonstandard methods of encoding characters in the URL string.

The query portion of the URL is often used to submit data to the server. URL-encoding is a technique defined in the URL/URI specifications for mapping 8-bit data to the subset of the US-ASCII character set allowed in a URL/URI. Without proper validation, URL-encoded input can be used to disguise malicious code for use in a variety of attacks.

Common Network-Level Exploits

While web application attacks are a new and growing threat, network layer attacks have long been a concern. Any practical approach to security must take into consideration how to defend against attacks at *all* layers. Some of the more common network-level exploits are described briefly below.

Denial of Service (DoS) Attacks

A denial of service attack is launched when an attacker overloads a site's resources until service slows or stops. This type of attack is carried out by directing a flood of connections to the site at such a large scale that it surpasses the network's ability to offer services to other users. The objective of a DoS attack is to disrupt access to the site's information and services.

One example of this is a SYN Flood, or TCP SYN attack. In this case, the attack begins with the client sending a large number of SYN messages to the server. These requests appear to be legitimate but in fact reference a client system that is unable to respond to SYN-ACK messages. In this sequence the handshake is never completed, creating a series of half-open connections. As the number of half-open connections builds, the server eventually exhausts its resources and stops accepting any TCP connection requests,

resulting in denial of access to genuine users. Other well-known DoS attacks include WinNuke, teardrop, Land, and bonk.

Distributed denial of service (DDoS) attacks use a large number of attacking machines to stage a DoS attack against a victim. The attacker first compromises a number of machines, then coordinates their efforts to launch the attack on a larger, targeted victim. Tools for launching DDoS attacks include Stacheldraht, Tribe FloodNet (TFN), and Trinoo.

Betrayal of trust network layer attacks

IP spoofing and TCP session hijacking both involve subverting trusted or authenticated communication links. Potential results include loss of confidentiality and data corruption.

Other TCP stack attacks

Other attacks target specific components of the TCP stack. An attacker may fragment a TCP header in an attempt to bypass IDS systems, perhaps to prepare for future attacks. An IPFilter type attack sets spurious, unusual TCP flags set.

The list of potential attacks and kinds of attacks is large. Whether through IP infrastructure or the applications themselves, tracking and maintaining potential attacks and their defenses is an enormous, ongoing job

Covering Their Tracks

Computer systems of interest to hackers usually keep track of all authorized and unauthorized access attempts. Records, called computer logs, provide useful and often critical clues that a trained specialist can use as the starting point to trace the hack.

Sophisticated hackers understand this and alter the logs upon gaining unauthorized access, thereby hiding the evidence of their crimes.

The attacker will usually delete or alter the log files. Attackers might delete the logs before they leave the system or just edit out relevant parts. This can get tricky, as some events are logged as individuals leave the system. Most attackers will use a program that keeps the logs clean and make any changes necessary after the leaving the system.

Beyond the DMZ

For years companies have been implementing strategies to protect network resources from intrusion and corruption. By combining a variety of devices, companies have built reasonably good defenses against many network layer (2-4) attacks.

Web applications and services have changed the nature of the threat. Attacks at the application level can bypass traditional firewalls by embedding within HTTP traffic. As companies add ever-more critical

web-based services, the concept of the demilitarized zone (DMZ) is no longer adequate to protect corporate resources.

According to Gartner, organizations need to move beyond the concept of the DMZ. Instead of worrying about servers and databases, the focus should be on *business transactions* themselves, creating the concept of the Transaction Zone for security. Network firewalls and VPNs are not enough; the new security approach must protect data and sites from application-based attacks, and incorporate secure logging for all web-based transactions.

The NC-1000 offers next-generation web security by creating a web security gateway that consolidates network-level attack prevention while defending many potential application-based attacks. The web security gateway shields all servers and resources behind it from scans, probes, and direct attacks, and filters potential attacks within HTTP traffic. The NC-1000 can fit into the current security architecture, with DMZ, while supporting the evolution to a Transaction Zone approach to security.

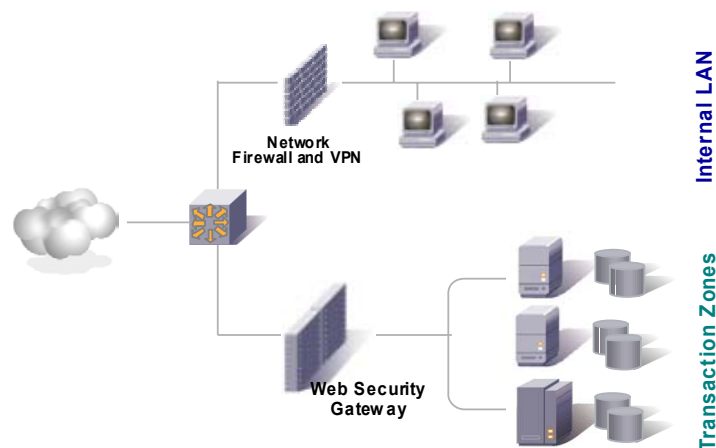


Figure 2: The NC-1000 can protect a new transaction zone for web applications and services

A Practical Approach to Web Security

As we've noted before, there is no single, perfect solution for web security. However, there is a better approach than today's reactive, patch-driven strategies that require a high degree of vigilance and developer training.

The NC-1000 creates a new line of defense at the perimeter of the web site. All external access, whether employees, partners, or customers, goes through the NC-1000. With centralized policy administration and management, network administrators can ensure that policies are enforced consistently across the enterprise.

This proactive approach differs from traditional web security in a number of ways:

- Instead of pursuing each potential weakness in every application throughout the site, the web security gateway offers a single location to implement security and screen attacks, through which all traffic must travel.
- It's good practice to maintain vendor patches for software products, but experience tells us that even large, sophisticated companies frequently fail to do so. The NC-1000 offers another layer of protection by blocking many attacks before they can reach the affected software programs.
- Instead of allowing all traffic then looking for potential attack signatures, the NC-1000 supports a "Deny unless allowed" policy.

This proactive approach means that the NC-1000 offers an instant and consistent level of security for both known and unknown attacks.

As we've already indicated, no single product or approach can protect you from all potential attacks. But by taking these three steps, the NC-1000 protects your site against a great majority of potential threats.

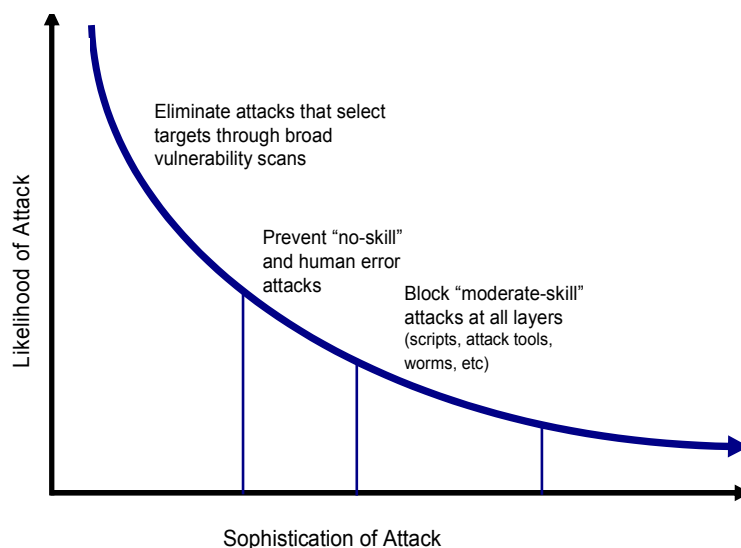


Figure 3: A typical risk profile for a site; the easiest attacks are most likely to occur, while those requiring more skill on the part of the attacker are less common. All can be enormously disruptive or damaging.

Attackers work by scanning for problems, exploiting weaknesses, and covering their tracks. The NC-1000 eliminates thwarts these activities by:

1. Cloaking the site from scans and probes;

2. Blocking common application- and network-level attacks;
3. Ensuring that web logs remain intact, so attackers cannot cover their tracks.

Step 1: Cloak the Site

The first step is to prevent attackers from searching the site and retrieving information such as OS and web server versions, directories, etc. The NC-1000, as a full proxy server, terminates all TCP connections, thus blocking the servers behind it from scans. It doesn't create connections to the servers behind it until a valid HTTP request is received. And because it uses a high-performance ASIC created specifically for TCP termination and SSL encryption, it can fill this role without degrading site performance.

The NC-1000 shields the site in other ways, such as masking all headers, blocking URL return codes, and keeping hidden directories private through the use of start URLs. The NC-10000 can inspect the HTTP payload and drop potentially malicious connections without passing them to the server at all.

By simply blocking the site from active probes and scans, the NC-1000 thwarts individuals simply searching for easy ways to launch an attack. The NC-1000's role as a proxy even blocks break-ins due to misconfigured servers (such as forgetting to restrict anonymous telnet access, or to change a system account from defaults).

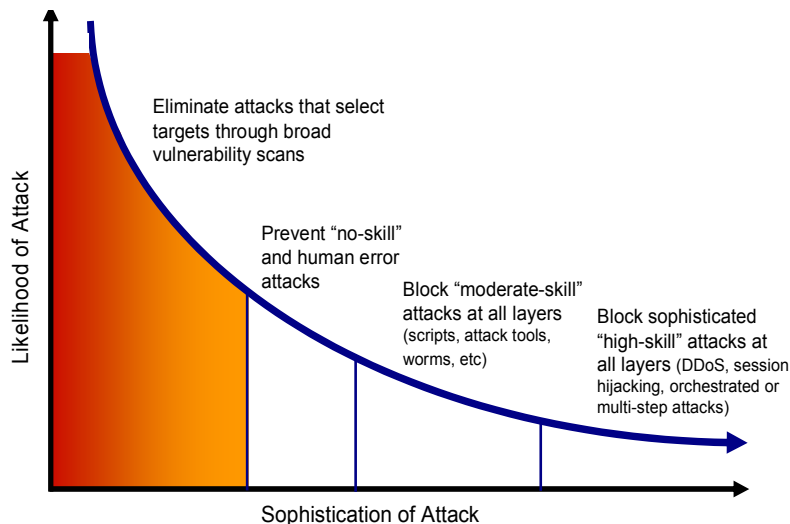


Figure 4: Cloaking the site immediately reduces the threat profile by eliminating a large number of "easy" attacks discovered through automating scans.

Step 2: Block Attacks at All Layers

The NC-1000 has a number of features which protect you from both network-level and application-specific attacks.

Application-layer attacks

The NC-1000 is not a replacement for good coding practices. For example, input validation for forms really should occur at the application level; no external device can have the application-specific knowledge to adequately protect applications from all potential corruptions.

However, the NC-1000 prevents many application-layer attacks with the following features:

HTTP header and URL normalization (autocorrect): The NC-1000 puts headers and URLs into a standard canonical format, dropping spaces, meta-characters, and other malformed constructions. This excludes attacks using constructions like “//” or “../..” that could allow access to forbidden server resources. By decoding the URL encoding or Unicode before filtering keywords, the NC-1000 prevents disguised input attacks.

URL length limit: Administrators can define a URL length limit, blocking excessively long URLs that are characteristic of buffer overflow attacks. A URL length limit would prevent both Nimda and Code Red attacks.

URL and Header blocking: The NC-1000 screens for preconfigured keywords in URL queries and HTTP headers. To protect against system command attacks, for example, you would filter for the appropriate command, such as open(), sysopen(), etc. Likewise, you can filter meta characters that could be used to launch system commands or take control of scripts. Scripting, using in cross-site scripting attacks, is relatively easy to filter out because of its meta characters and script commands.

Dynamic Access Control List: The NC-1000 administrator creates a defined set of starting URLs. The NC-1000 parses all the URLs in the server responses, creating and maintaining a dynamic Access Control List for allowed URLs. This prevents forceful browsing.

Cookie security: By encrypting and digitally signing cookies, the NC-1000 prevents input validation attacks launched in cookies (cookie poisoning).

Network layer-attack prevention

Full termination and proxy: Because the NC-1000 terminates all TCP sessions, it blocks IP fragmentation attacks, and prevents backend servers from being used as slaves to launch DDoS attacks. It doesn't pass other protocols through to servers, so unknown protocols and UDP bombs will not affect the NC-1000.

Port restriction: The NC-1000 only allows traffic to ports 80 and 443; this eliminates many attacks launched at other applications and services, including buffer overflows and specific service attacks.

Threshold client rate limiting: The NC-1000 is designed to prevent Denial of Service attacks by dropping or throttling unwanted traffic while maintaining connections from legitimate clients. The NC-1000 supports ICMP rate-limiting on a service-by-service basis. It uses TCP windowing technology to resize windows once predefined thresholds are reached, without dropping data.

Application-layer VPN: With integrated, on-chip SSL capabilities and built-in certificate authority, administrators can use SSL to create application-layer VPN to specific applications.

Random sequencing TCP session numbers – On-chip random number generation prevents TCP session hijacking by making it impossible to guess sequencing numbers.

VLAN firewalls – Administrators can create virtual LAN firewalls, so that a worm infecting one part of the network cannot hop to servers on another VLAN.

NAT and half NAT– because it provides Network Address Translation and fully terminates TCP sessions, potential attackers cannot see the IP address of servers.

By blocking common application- and network-layer exploits, the NC-1000 further reduces risk to valuable web services and data.

Step 3: Prevent Log Tampering

The NC-1000 keeps two types of logs: web logs and sys logs. The sys log essentially tracks all events occurring against the NC-1000 itself. Web logs (also called transaction logs) track details occurring against production traffic.

The sys log is maintained on the Linux management system, stored as a set of files on the disk drive. Administrators can optionally send the log via management port to remote daemons, and can configure the detail level of tracked data.

The web log can be configured for each service and logs information about production traffic for that service. These web logs track:

- The client's IP address
- The client's authentication name
- Date and time of the request
- A copy of the request exactly as it came from the client
- The HTTP status code returned to the client
- The content length returned in bytes

Logs may be kept in W3C (CLF) or Squid formats. Logs may be digitally signed, encrypted, or both before being transferred via FTP to a web log server. Because logs are stored separately from the individual web or database servers, even accessing the log is difficult for an intruder.

Digital signatures ensure that a log has not been altered since its creation, providing non-repudiation as well as enhanced security. Encryption ensures that no one can view the contents of the log except the owner of the encryption key. Encrypted, signed logs cannot be modified, even by internal administrators. If a web log is deleted, the secure, signed administrative log will record the action.

Summary

The number of potential threats to web sites today is too numerous to track, and growing almost daily. Although network-layer attacks are relatively well understood, they still remain a challenge. At the same time, a new class of application-layer attacks is using the HTTP payload itself, proving very difficult for traditional firewalls and other approaches to block.

Relying solely on best practices in application development as a means to secure applications is not practical. Many developers lack adequate training in security issues and inadvertently leave openings in applications. Commercial software is vulnerable as well; vendors are continually releasing security patches to address newly discovered problems. Keeping patches up-to-date is a continuous struggle.

The NetContinuum NC-1000 offers a new, practical approach to web security that incorporates both network-layer and application-layer protection. As a full proxy server, the NC-1000 shields web sites from scans and probes – eliminating a great many potential attackers using automating scanning techniques to look for targets. Other features examine the HTTP payload for potential attacks; configurable filters can be added as new attacks become public.

By consolidating security efforts and processes that today are distributed throughout the enterprise, the NC-1000 improves site security while reducing its cost. A proactive security stance not only blocks known attacks, but shelters sites from potential new exploits that will inevitably be discovered over time. It also provides some protection from simple human error, such as improper configurations, that opportunistic attackers often discover and exploit.

With a next-generation web security gateway, organizations can continue to leverage the benefits of web-based applications and services while containing the risk to valuable customer relationships, privacy, and data integrity.

