



Sidewinder® G2 Firewall™ Type Enforcement® Technology

SECURE COMPUTING

Secure Computing Corporation Corporate Headquarters

4810 HARWOOD ROAD
SAN JOSE, CA 95124 USA
TEL +1.800.379.4944
TEL +1.408.979.6100
FAX +1.408.979.6501

European Headquarters

EAST WING, PIPER HOUSE
HATCH LANE
WINDSOR SL4 3QP UK
TEL +44.1753.410900
FAX +44.1753.410901

Asia/Pac Headquarters

801 YUE XIU BLDG.
NOS. 160-174 LOCKHART RD.
WANCHAI HONG KONG
TEL +852.2520.2422
FAX +852.2587.1333

Japan Headquarters

LEVEL 15 JT BLDG.
2-2-1 TORANOMON MINATO-KU
TOKYO 105-0001 JAPAN
TEL +81.3.5114.8224
FAX +81.3.5114.8226

www.securecomputing.com

TABLE OF CONTENTS

Sidewinder® G₂ Firewall™ Type Enforcement® technology

Introduction3
Overview of Type Enforcement® technology3
Using Type Enforcement technology in Sidewinder G ₂ Firewall . . .5	
Network stack separation5
Triggers: detecting intruders6
Control for “super users”7
Principle of least privilege and controlled system calls7
Implementing Type Enforcement technology8
Domains and types8
Conclusion10



Introduction

Business and government organizations of all sizes have embraced the Internet as a preferred medium for conducting mission-critical operations. One significant factor these organizations have had to address in this e-business evolution is an unprecedented demand for security to ensure the confidentiality, integrity, and availability of their critical data and services. The best starting point for building reliable security that protects a company's assets is to install network perimeter-based protection with powerful capabilities that match the security needs of the organization.

Secure Computing Corporation is the provider of the strongest network security gateway solutions available today. Secure Computing has achieved this level of reliability by combining application-layer gateway and VPN technology with our own patented Type Enforcement® technology. Type Enforcement technology is a critical element of the SecureOS™ operating system, on which Secure Computing's Sidewinder® G₂ Firewall™ access gateways operate.

Unlike the access controls in most operating systems, Type Enforcement technology is a mandatory access control mechanism. That is, Type Enforcement controls are based on factors that can only be changed by the system security administrator. If a hacker were to gain access to a G₂ Firewall system, he would not be able to change the Type Enforcement controls. By their very nature are the first security system a hacker encounters in the corporate network. If a hacker were to succeed in compromising the firewall, he or she would have an open door to the corporate network. In this paper, we describe Type Enforcement technology, what makes it work, how it relates to the principle of *least privilege*, and how it works with our G₂ Firewall.

Overview of Type Enforcement® technology

Secure Computing developed Type Enforcement technology to provide integrity for critical government computing systems targeted at the highest level (A1) of the U.S. Government's Trusted Computing Systems Evaluation Criteria. Since then, Type Enforcement technology has evolved into a powerful mechanism for protecting the integrity of any mission-critical system, including network firewalls. Type Enforcement technology goes beyond the security mechanisms in commercial operating systems by eliminating unconstrained privileges and extraneous services. Commercial operating systems are designed for convenience, not security. This may appear to make application development or administration easier. In fact, it makes both jobs more difficult because it requires both developers and administrators to be much more security aware as they write their code or configure their systems. Their jobs are much easier when the operating system provides strong underlying security mechanisms.

The Type Enforcement feature of SecureOS provides strong separation of:

1. the operating system from applications
2. applications from each other

This characteristic, unique to Type Enforcement technology, is the core of the robust security provided by the Sidewinder G₂ Firewall.

With its Type Enforcement security mechanism, the G₂ Firewall resembles a honeycomb where critical system components are placed in separate cells. As part of G₂ Firewall's Type Enforcement technique, each process is confined to a cell where it can only access system resources, such as a files, sockets, and directories, that it needs to do its job. In particular, the Type Enforcement mechanism controls what code the process can execute. This level of control completely eliminates the possibility of stack overruns, the major form of firewall attacks. In 1998, nine of the thirteen firewall advisories published by CMU's Computer Emergency Response Team (CERT) were stack overrun attacks.

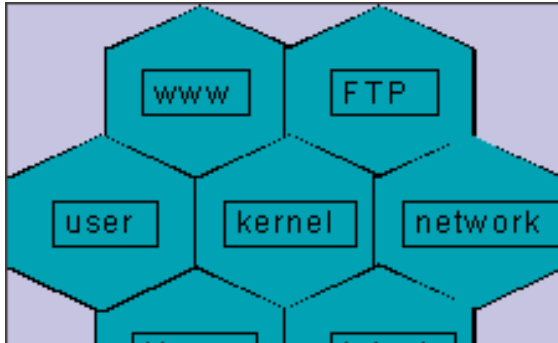


Figure 1. Type Enforcement technology provides a structure that separates applications and controls which applications users can access. A file or application program must be in the same cell as a process for the process to access it.

SecureOS also enforces restrictions on the cells in which each user can have active processes, based on the user's role in the system. Thus, each user can only run the processes he needs to run to do his job. Since the Type Enforcement modifications in G₂ Firewall's SecureOS have been made at a low level in the UNIX kernel, even when a process is running as root (the lowest level possible), it is constrained by Type Enforcement boundaries. If an attacker does manage to break through, Type Enforcement would constrain them into a single "cell." Furthermore, in SecureOS's operational mode, the concept of a super-user with "root access" does not exist, so the possibility of a hacker breaking in, fooling the system into thinking he or she is an administrator with super-user privileges, and gaining unlimited access to the system is not possible. In SecureOS, the "root" account has no special privileges. The admin role operating in the Admin domain has access to most system files, but is still not as powerful (or dangerous) as root on a standard UNIX system.

So to compromise the Sidewinder G₂ Firewall, a hacker must bypass both UNIX and Type Enforcement restrictions—an event which at the time of this writing has never occurred successfully. Even compromising UNIX is more difficult on the G₂ Firewall, since the honeycomb structure allows us to place critical configuration files and UNIX tools out of hackers' reach.

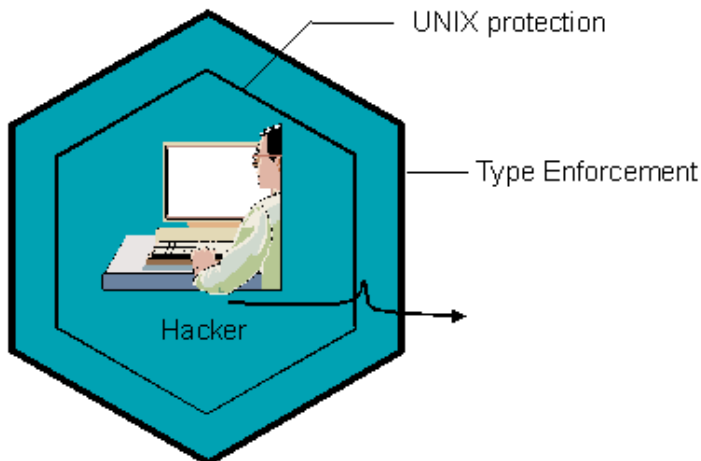


Figure 2. To break out of a Sidewinder G₂ Firewall cell, the hacker must bypass both the UNIX protection mechanisms and Type Enforcement barriers.

Using Type Enforcement technology in Sidewinder G₂ Firewall

Sidewinder G₂ Firewall uses the Type Enforcement mechanism in SecureOS to provide a number of security features, including:

- Network stack separation
- Triggers for intrusion detection
- Control of “super user” privileges
- Principle of *least privilege*

In this section we describe each of these.

Network stack separation

Sidewinder G₂™ architecture takes full advantage of the Type Enforcement mechanism to compartmentalize the system, much the way that the bulkheads in a ship separate the hold into compartments so that a break in one compartment does not result in the complete loss of function, or in the compromise of other compartments.

Network stack separation provides a good example of this. One of the goals of a firewall is to keep information coming from the Internet separate from information coming from the internal network. If a firewall does not have network stack separation, all the network packets come up the same stack, all mingled together. In this case,

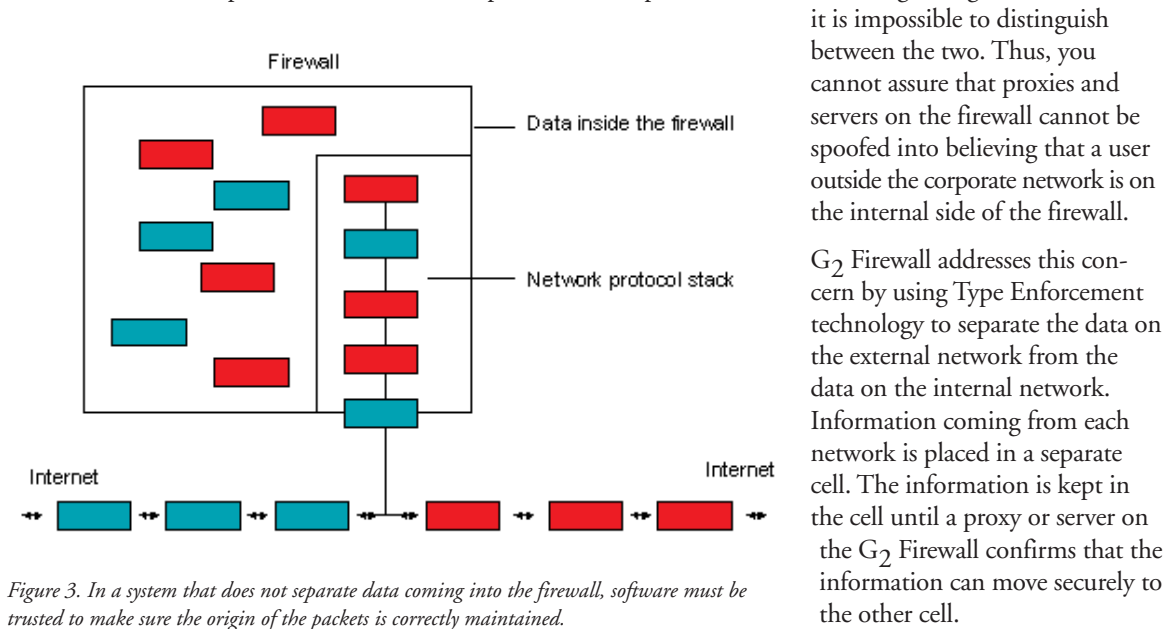


Figure 3. In a system that does not separate data coming into the firewall, software must be trusted to make sure the origin of the packets is correctly maintained.

With this architecture, G₂ Firewall can be compared to a bridge connecting the corporate network on one bank with the Internet on the other bank. As shown in Figure 5, each network service, e-mail, Web, database access, video conferencing, etc., has its own lane across the bridge. Type Enforcement technology separates the lanes so that traffic in the Telnet lane cannot jump into the HTTP or FTP lanes where it could do immeasurable damage. If a hacker were to compromise a service on the G₂ Firewall, he could not take advantage of this to do the things hackers like to do most:

Sidewinder® G₂ Firewall™ Type Enforcement® technology

- launch an attack on other services
- modify the underlying security policy
- launch a sustained attack by importing attack tools onto the G₂ Firewall

Other services and security policy maintenance in general are functions of cells the hacker (as well as other users) cannot access. All users are constrained to running processes that execute only the appropriate code for the cell. In particular, hackers and/or users cannot use the processes to execute their attack tools.

Triggers: Detecting intruders

Sidewinder G₂ Firewall also uses Type Enforcement technology to detect and respond to malicious intruders. On the G₂ Firewall, a partial model of malicious behavior is defined ahead of time. The model is represented by placing triggers on specific files that normal user processes do not need to access, such as system configuration files or the master password file. (Because triggers are specific to Type Enforcement cells, administrators who need access to files with triggers can be placed in a different cell that does not set off the trigger.) If an unauthorized process tries to access a file that contains a trigger, the G₂ Firewall can contact the system administrator by pager, e-mail, or an SNMP alarm. Because triggers are added to the G₂ Firewall as part of the kernel, they cannot be bypassed.

By creating application specific software, other actions can be taken when a trigger file is touched. For example, the G₂ Firewall Challenge system logs the user out after a certain number of Type Enforcement violations.

G₂ Firewall's trigger approach for detecting malicious users offers several advantages:

- Malicious behavior is clearly defined.
- As a result, 100% of all defined malicious behavior is detected.
- The kernel itself performs detection, not a separate audit system.
- Malicious users cannot bypass or shut off the detection mechanism. The reaction to the event occurs in a separate subsystem running in a cell that users cannot access.
- The malicious behavior model is static. Because the model cannot be modified while the system is running, malicious users cannot change how malicious behavior is defined on the system.

The trigger approach is further strengthened by Type Enforcement technology because users operate in cells that give them access to only the files they need. So even if a user is malicious, the effects of his or her behavior are limited. By placing a few tempting files within a user's domain, malicious users can quickly be detected, as shown in Figure 5.

In addition to file triggers designed to catch malicious users, administrators can also configure application-specific triggers for detecting malicious behavior within an application.

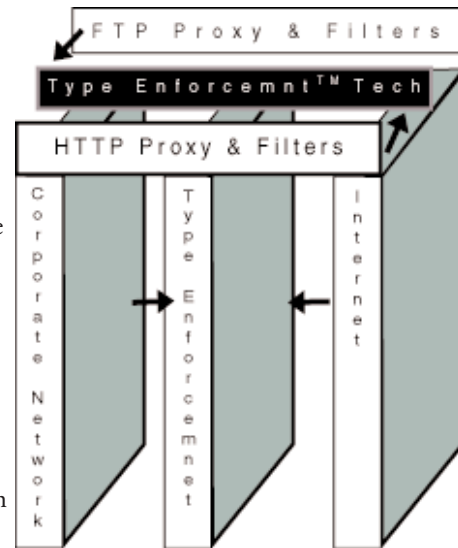


Figure 4. Application proxies are like separate lanes of a bridge connecting the corporate network to the internet. Each lane is configured to enforce appropriate rules for its traffic. Type Enforcement provides strong barriers to separate the traffic lanes.

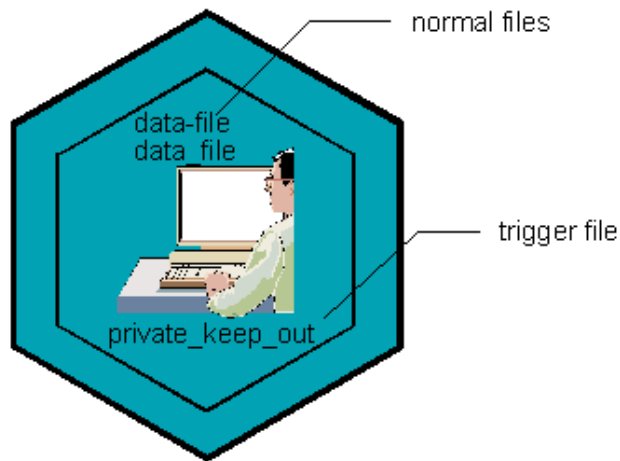


Figure 5. To detect malicious users, the administrator places a few tempting files within a user's security perimeter. These files have the trigger flag set, which the kernel checks when the file is accessed. The response to hitting a trigger file can be application specific.

Control of “super users”

Most operating systems have some form of “super user” privilege. On UNIX systems it is called root. Windows NT/2000 has two levels of super user: system and admin. Processes running with super user privileges can change security settings on the system. These settings include file ownership or accesses, the ability to turn off audit, etc. While this may afford the administrator and application developer some convenience, it poses a considerable security risk if the super-user privilege were compromised. Attackers who successfully break in often manage to convince the system that they are the super users—which gives them complete control over the entire system.

SecureOS closes up this loophole, since the Type Enforcement control applies even to processes with super user status. That is, even a process with super user status is confined to its own cell.

Principle of least privilege and controlled system calls

By using Type Enforcement technology to confine each process to a specific cell, SecureOS enforces what security engineers call the principle of *least privilege*, which mandates that each process should have access to those resources required to do its specific task, and nothing more. With the Type Enforcement honeycomb, each application or service (e-mail, Telnet, FTP, etc.) is separated from the others with impenetrable virtual barriers between them. This prevents vulnerabilities that would, for example, allow an attacker to make use of the HTTP (www) protocol to carry out an attack on other services.

Type Enforcement technology does even more to enforce the principle of least privilege. Operating systems have many privileged system calls that malicious users can use to access the kernel directly and compromise the system. Type Enforcement technology prevents this sort of attack by associating a series of special flags for each cell. These flags indicate which system calls can be made from that cell. For example, certain system calls are only permitted in cells that are restricted to system administrators and their processes. Even root access will not allow a process to make disallowed calls. Each cell is allowed to make only the system calls required for processes and users in that cell to do their jobs.

With Type Enforcement, less trusted users, or processes executing questionable code, can be isolated so that they cannot make any privileged system calls and can only access files and other resources that are not critical to system operation.

Implementing Type Enforcement technology

To implement Type Enforcement technology for SecureOS, Secure Computing has modified a standard BSD/os UNIX kernel. In this section we explain the modifications to the kernel and show how they are used to create the Type Enforcement cells we described above.

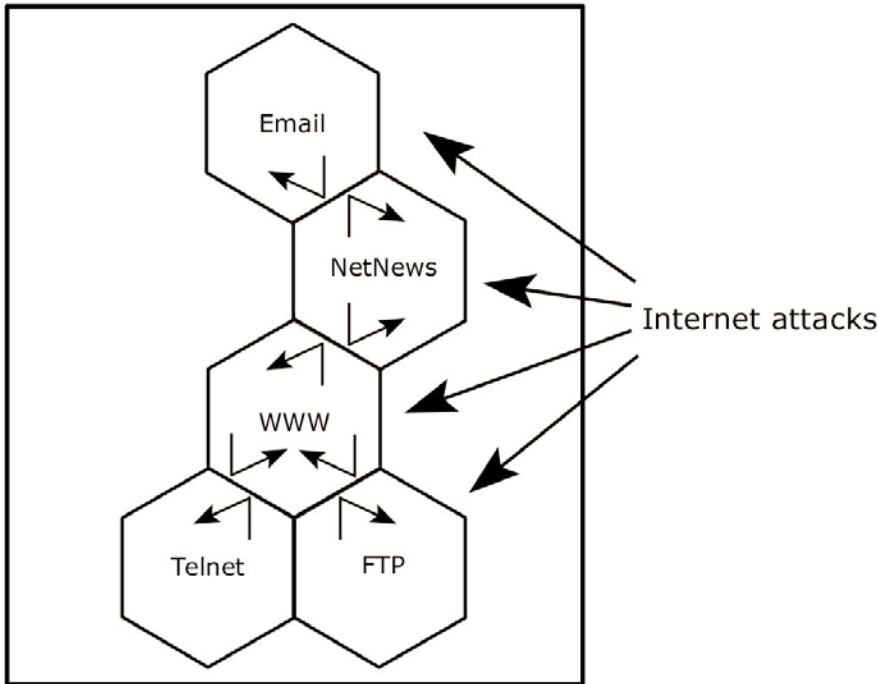


Figure 6. Sidewinder G₂ Firewall's separate service domains. Attack attempts are constrained to their particular cell.

Domains and types

SecureOS associates a domain attribute (or label) with each process and type attribute (or label) with each system object (file, directory, socket, etc.). These labels are functional labels. That is, the domain attribute indicates the intended function of the corresponding process, and the type attribute indicates the purpose of the object. The trick then, is to add an access control mechanism to the system that consults the appropriate labels before permitting a process any access to an object and to only allow those accesses required so that:

- processes can perform their functions
- objects are maintained so they can fulfill their intended purposes

SecureOS stores the minimal set of required accesses in the system Domain Definition Table (DDT). This is a table indexed by domains and types. The entry corresponding to a particular domain and type indicates the set of accesses needed by processes in the domain, to perform that process's intended functions. The DDT also contains a per-domain entry indicating the privileges permitted to processes in the domain. The set of accesses depends on the semantics of the object type. File types support create/delete, read, write, and execute accesses. For instance, socket types support create/delete, connect, and bind accesses. File types support create/delete, read, write, and execute accesses.

The distinction between file execute and file read access provides control of executables, which is an important element of Type Enforcement technology. Many of the favorite hacker attacks involve tricking a process into executing code that the attacker wrote. (Stack overrun attacks are the best known example of these, as stated on page 4.) In an HTTP overrun attack, the attacker feeds the Web server a URL that fills up the HTTP process's stack and overflows into memory space that contains the attacker's code. This causes the Web server to run this code rather than the intended Web server code. Type Enforcement technology can prevent overrun attacks and other forms of executable spoofing/replacement attacks.

Because the system refers to the DDT at every critical system call, Type Enforcement barriers cannot be circumvented.

In practice there is often one set of domains and types for each application subsystem, because each application has different access requirements. A process is said to be in the domain assigned to it. Each process in a given domain is set up to handle one kind of application, and each application runs in its own set of domains. In effect, each Internet application or service "lives" in a separate area. These applications and services cannot access data from another area, unless explicitly allowed by the Type Enforcement policy. Any such "pipelines" between applications can be tightly controlled.

This fine-grained level of control provides the honeycomb previously described on page 4. It restricts the ability of a hacker to use an application to attack the rest of the system. Examples of process domains are execution environments for applications such as an FTP or Telnet proxy. The Type Enforcement policy can be configured to eliminate the possibility of any cross talk or interference between the two.

The figure below illustrates how Type Enforcement technology controls a domain's access to files of different types. Any time a process tries to access a file, the Type Enforcement controls determine whether the access should be granted; these controls cannot be circumvented.

In the diagram below, please note the following points:

- A process running in Domain A is attempting to access File Type X.

Type Enforcement technology denies this request.

- A process in domain B is permitted access to File Type X and File Type Z, concurrently while
- The process in domain C is granted access to File Type Y.

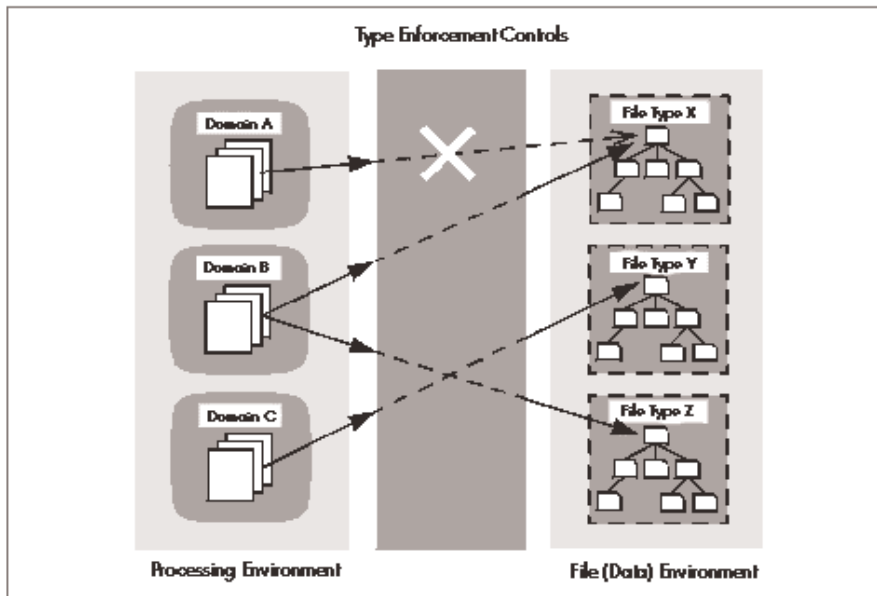


Figure 7. Type Enforcement divides a Sidewinder G₂ Firewall system into domains and file types.

Conclusion

At Secure Computing, we use Type Enforcement technology to protect the integrity of our Sidewinder G₂ Firewalls access gateways. We do this by assigning separate domains and types to each of the application proxy subsystems (i.e., e-mail, Web, SQL, audio, etc.) as well as to each of the networks protected by the firewall. This separation of the network services creates virtual chasms between the networks that only the pre-defined proxies and filters are permitted to bridge. No other process, not even a root process, can read from or write to the networks.

Moreover, root access and extraneous services are eliminated by SecureOS. These restrictions assure that G₂ Firewall can protect itself against attack—something that cannot be claimed by other security solutions running on commercial operating systems. Thus Type Enforcement technology is a critical part of the SecureOS operating system. It removes the inherent risks of layering an access gateway on top of a commercial operating system. Also, the fact that SecureOS is embedded in Secure Computing's G₂ Firewall results in a significant cost savings. A G₂ Firewall customer need only purchase and install a single network access system, rather than first purchase and install an operating system and then a network access system. Similarly, maintenance and policy management are cut in half.

The strength of Type Enforcement technology has been demonstrated in many different venues. For the last six years, Sidewinder technology has withstood over 10,000 attacks on the Sidewinder challenge site sponsored by Secure Computing Corporation without break-in. Sidewinder G₂ Firewall delivers the strongest perimeter application protection available to organizations of all sizes, from small companies and departments up to Fortune 50 enterprises and the U.S. military.

The combination of an application-layer gateway with Type Enforcement technology provides a one-two punch far stronger than is available in any traditional firewall or gateway product running on an unmodified commercial operating system, *no matter how well it is configured*. Furthermore, by extending the powerful principles of Type Enforcement technology—least privilege, containment, operating system integrity, intrusion triggers, and control of privilege—into new areas, Secure Computing has created a superior architecture for comprehensive security for the extended enterprise.