NOKIA

SecureXL and Nokia IPSO

White Paper

1 April 2008

Customer Confidential

Contents

1 What is SecureXL?

SecureXL is the security performance architecture of Check Point VPN-1 Power (Check Point integrated firewall, VPN, and intrusion prevention solution) and Nokia security appliances. The architecture offloads many intensive security operations to optimized Nokia IPSO code running on Intel x86 hardware or on network processor hardware. Offloaded security operations include TCP state negotiation, packet forwarding, Network Address Translation, VPN cryptography, anti-spoofing, routing, and accounting. Optimized IPSO code placed at the hardware interrupt level or in a network processor reduces the overhead involved in performing these security operations.

1.1 What Does It Do?

SecureXL accelerates firewall and VPN performance by remembering certain attributes of packets and packet flows that have already been validated by the firewall/VPN application. Thereafter, validation of related packets and connections is delegated to IPSO across the SecureXL API. IPSO either performs this validation natively at the hardware interrupt level on x86 hardware or supervises execution of further optimized code in attached network processors in IP security appliances that support them. Both of these approaches involve substantially less computing overhead than required by the firewall/VPN application itself.

The rest of this white paper explains the details of how and why SecureXL accelerates firewall and VPN throughput and connection rate performance, what features have been added to previous SecureXL versions, what features are in SecureXL 2.22 (currently implemented in IPSO 4.2), and what its requirements and limitations are.

2 Firewall Flows and SecureXL

Like Nokia IPSO's "Firewall Flows" feature, SecureXL reduces the overhead involved in forwarding packets that are parts of flows which the firewall has already validated. SecureXL replaces, in part, the Firewall Flows feature introduced in earlier versions of IPSO.

Unlike Firewall Flows, SecureXL extends this acceleration to firewall traffic connection rate and to encrypted VPN traffic throughput as well.

Note that SecureXL makes use of the infrastructure provided by and operates on top of firewall flows. SecureXL is not mutually exclusive to firewall flows, but actually needs firewall flows mode to be operational in order to be used. Also, IPSO's slow path is not used with SecureXL.

	Firewall Flows	SecureXL
Accelerate firewall (unencrypted) traffic throughput	Yes	Yes
Accelerate firewall (unencrypted) traffic connection rate*	No	No
Accelerate VPN (encrypted) traffic throughput	No	Yes

** Note that connection rate is accelerated by SecureXL only when Network Address Translation (NAT) is disabled. Additionally, VPN (encrypted) packets are not connection rate-accelerated.*

3 Throughput Acceleration

Packets attempting to establish a new TCP connection (or a comparable UDP connection table entry in the firewall) are handled in the slowpath. Once the first packet is seen by the firewall and suitable connections/flows information is offloaded to IPSO, further packets are handled at IPSO's interrupt-level code.

The round-trip processing- interrupt driver to application back to driver-level code – is very time consuming and relative to the minimal processing necessary for later packets (that can be done entirely at the driver level). Those later packets, determined to be part of an existing, already validated flow, are forwarded directly from the driver level without the overhead of firewall application involvement.

SecureXL improved non-encrypted firewall traffic throughput, and encrypted VPN traffic throughput, by nearly an order-of-magnitude- particularly for small packets flowing in long duration connections.

Following the in depth description of how SecureXL accelerates connection establishment below, we'll illustrate the high-level flow of packets through the Nokia SecureXL-accelerated security appliance.

4 Connection Rate Acceleration

SecureXL provides another form of acceleration. It reduces the overhead in establishing certain kinds of new connections, improving new connection rate (connections per second) and connection setup/teardown rate (sessions per second) as well as throughput in certain high-connection-rate traffic environments.

The principle involved is a simple extension of the Firewall Flows and SecureXL approach to "one-time validation" of a flow. The one-time validation is extended from a particular 5-tuple – source address, destination address, source port, destination port, and protocol (one classic definition of a

"flow", or the definition of a "microflow" by Internet researchers) – to a range, or block, of one or more of these "tuples". Specifically, "source port" of a flow may be masked off, effectively providing a global match for source port. That is, once a flow is validated and established, a "template" of that flow, with source port masked off creating a global match is saved and remembered (with a configurable timeout). Any new connection setup that matches four of the 5-tuples is again handled in the slowpath and not at the driver level. All new connection creation/old connection deletion is handled in the slowpath. However, these new connection setup packets matching 4 out of 5 tuples avoid a round trip to the firewall application, thus avoiding the computing overhead. Security is not impacted because IPSO continues to track the state of the new connection using stateful inspection.

5 Masking the Source Port – Creating a Global Match

Below we examine how ports are used in establishing TCP connections to understand how source port masking is particularly effective in reducing connection establishment overhead, and accelerating connection rate and throughput in certain high-connection-rate environments.

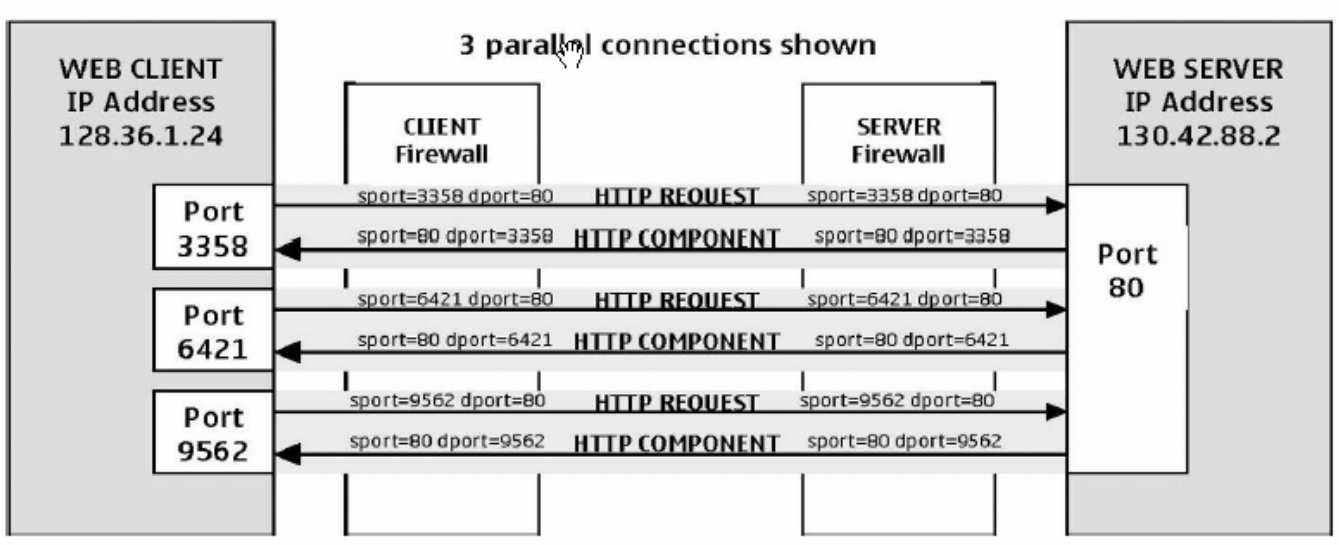
A client requesting a connection to a server will initiate the TCP three-way handshake. The client addresses the server typically at a well-known port number depending on the service provided by the server (e.g. port 23 for Telnet, port 80 for HTTP). Together, the server's IP address and the well-known port number form a "socket address". The client assigns and pairs an operation system selected port number with the client's IP address to create a socket address for the reverse direction.

6 It's all about the Application Layer Protocol

Now consider higher-level application protocols that involve numerous TCP connections between the client and server – either simultaneously (in parallel) or sequentially, or both. One of these protocols, which accounts for most Internet traffic, is HTTP. SMTP is another example, but we'll look at HTTP as representative of how SecureXL accelerates connection rate. FTP would not benefit by source port masking. The FTP control connections are handled by the firewall and the data connections are handled by the firewall until it determines that direction of the data flow.

6.1 HTTP and SecureXL-Accelerating Firewall Connection Handling

Web pages consist of multiple HTTP components – text and perhaps dozens of graphic elements. Using HTTP 1.0, each component is downloaded from server to client using a separate TCP connection. This action involves substantial overhead in connection setup and teardown, and further overhead in protective-firewall connection tracking (firewalls at both ends).



In all cases in the figure above, between a Web Client and a Web Server, TCP connection establishment is initiated by the Web Client, which then sends and HTTP request. The Web Server responds by sending the HTTP component (text or graphic).

HTTP Request

→ Each of the packets from the Web Client that requests an HTTP component from the Web Server has the same source address, destination address, destination port (80), and protocol (HTTP). Only the source port, assigned by the Web Client's operation system, one per connection, differs in order to create unique socket addresses at the Client for each HTTP request/component (via separate TCP connections for each component).

HTTP Component

← In going the other direction, each of the packets from the Web Server that build the web page components on the Web Client has the same source address, destination address, source port (80), and protocol (HTTP). Only the destination port differs (it's been assigned by the Client operating system to that connection).

Let's look at applying SecureXL at the SERVER Firewall. Once a connection involving a flow to port 80 is approved by the firewall application for the Web Client (resulting from the first HTTP request) a template is created and stored. All subsequent connection setups carrying those additional requests can share that same template "approval" because the fact that the source ports differ is okay. Establishing those subsequent connections does not involve a round trip to the firewall application, and those connections are created much more quickly through the SERVER Firewall as a result.

In the same way, at the CLIENT Firewall, once a connection involving a flow to port 80 is approved by the firewall application (as above), all subsequent connections carrying those additional requests can share that same "approval." Establishing those subsequent connections does not involve a round trip to the firewall application, and those connections are created much more quickly through the CLIENT Firewall as a result.

SecureXL accelerates subsequent connection establishment through both firewalls when multiple connections share the same source address, destination address, destination (server) port and protocol.

6.2 HTTP 1.0, 1.1

HTTP version 1.0 operates as described above, and SecureXL increases the connection establishment rate of the firewall for tracking these connections. This is because HTTP 1.0 creates a separate connection for each HTTP component. The newer HTTP version 1.1 improves the protocol's performance by permitting not only parallel, but also persistent and pipelined server connections. The server may keep the connection alive after sending the end of a component, which avoids the need to create a new connection to send the next component. HTTP 1.1 is supported by most web servers and the current generation of browsers as well.

High connection-rate network environments involve primarily HTTP traffic. While HTTP 1.1 is significantly less connection-intensive, HTTP likely remains the protocol that generates most new connection requests in enterprise and Internet traffic. At the same time, while overall traffic levels continue to grow, connection rates grow less quickly as network environments use primarily HTTP 1.1.

SecureXL connection templates create the opportunity to generate extremely impressive connection rate performance. Given the benefit of connection templates in heavy HTTP environments, the significant performance increase can, in fact, be reflected in real world traffic environments, particularly a Web Server farms, and in enterprises where there is a great deal of web traffic to a small, concentrated set of servers.

7 Effect on Other Application-Layer Protocols

The three main connection-oriented application protocols in use today are HTTP, SMTP, and FTP. HTTP behavior was described in detail above.

SMTP is typically transported over TCP. One or ore simultaneous connections can be opened to the SMTP server, and they may remain open for the transfer of multiple mail messages. Although the option exists to force a new TCP connection for each mail message, this is not normally done because of the overhead involved. So SMTP is not a connection-intensive protocol.

FTP typically involves long traffic streams and is not connection-intensive.

Other application-layer protocols- RPC, NFS, NNTP, NTP, are not so connection-intensive that they benefit from SecureXL templates.

Current trends are toward the increasing use of streaming protocols to carry audio and video programs, as well as Voice over IP and multimedia conferencing. These streaming protocols are not connection-intensive and benefit from SecureXL templates when streams are unidirectional.

8 UDP "Pseudo-Connections"

UDP is inherently connectionless. However, FireWall-1 tracks, and Firewall Flows and SecureXL accelerate UDP traffic through the firewall by tracking UDP pseudo-connections. Unlike TCP, applications that use UDP as their transport mechanism have no way of "closing" the pseudo-connections tracked by the firewall- the firewall lets them expire. There are no UDP-based, widely used, traffic-dominating applications that create high "pseudo-connection"-rates – that is, open multiple socket addresses fro short durations.

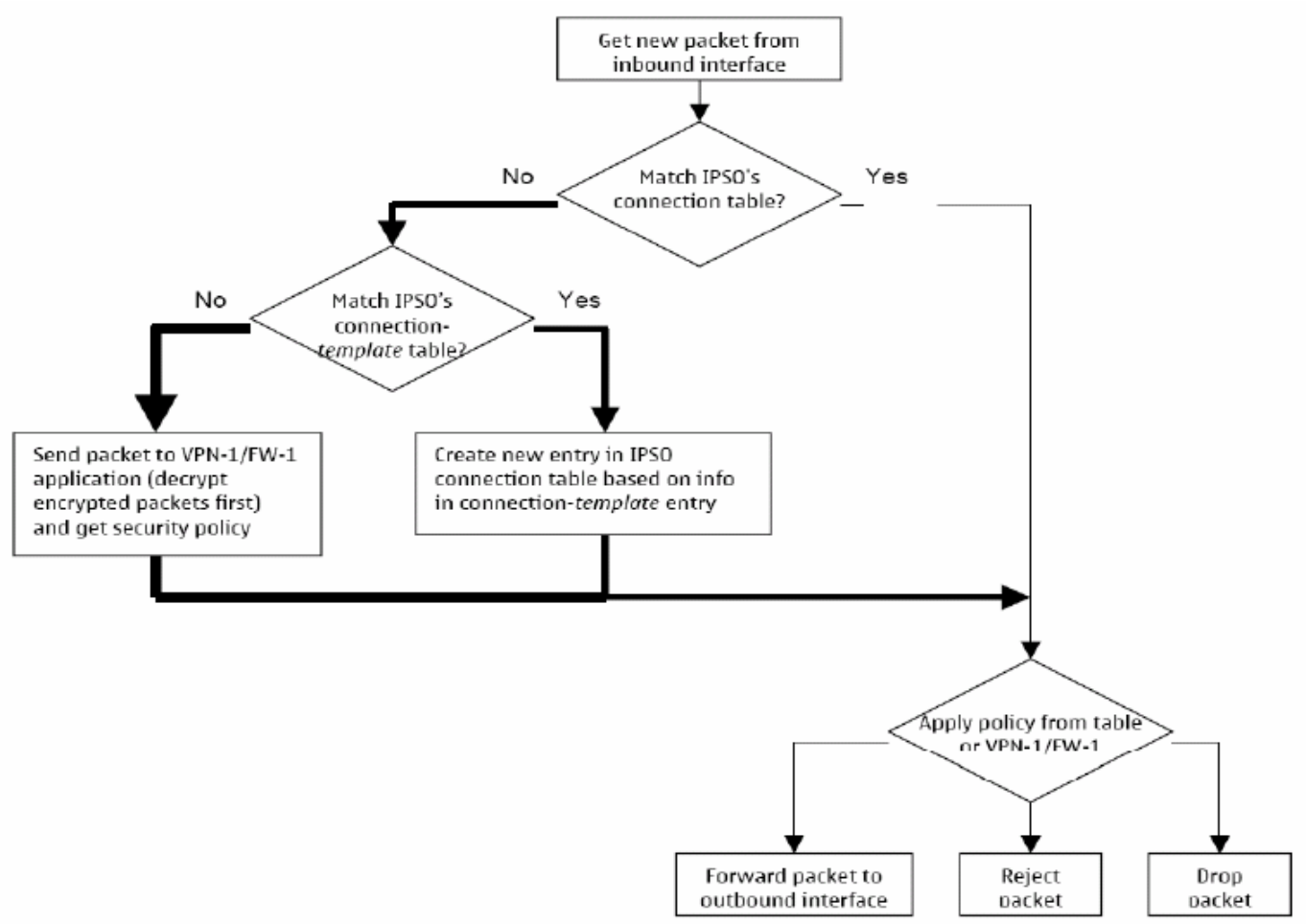
9 Packet Flow through SecureXL

The figure below shows the decision logic for packets flowing through the Nokia SecureXL-accelerated security appliances.

A new packet arrives at the inbound interface. The packet is checked against IPSO's connection table (which mirrors the VPN-1 Power connection table). If there is a 5-tuple match (src, dst, sport, dport, proto), then the new packet is part of an existing flow and is forwarded to the outbound interface for handling (forward, drop, or reject). This path involves the least amount of forwarding overhead and accelerates throughput for packets that are parts of an existing flow.

If the new packet does not match an entry in IPSO's connection table, then the new packet represents a new flow and requires a new connection table entry. However, if the packet matches an existing connection template, then the new connection table entry can be created (based on information in the connection-template table entry) without a round trip to the VPN-1 Power application. The packet is then forwarded to the outbound interface for handling. This path reduces the overhead involved in creating a new connection table entry and accelerates connection rate for new connections that match existing connection templates.

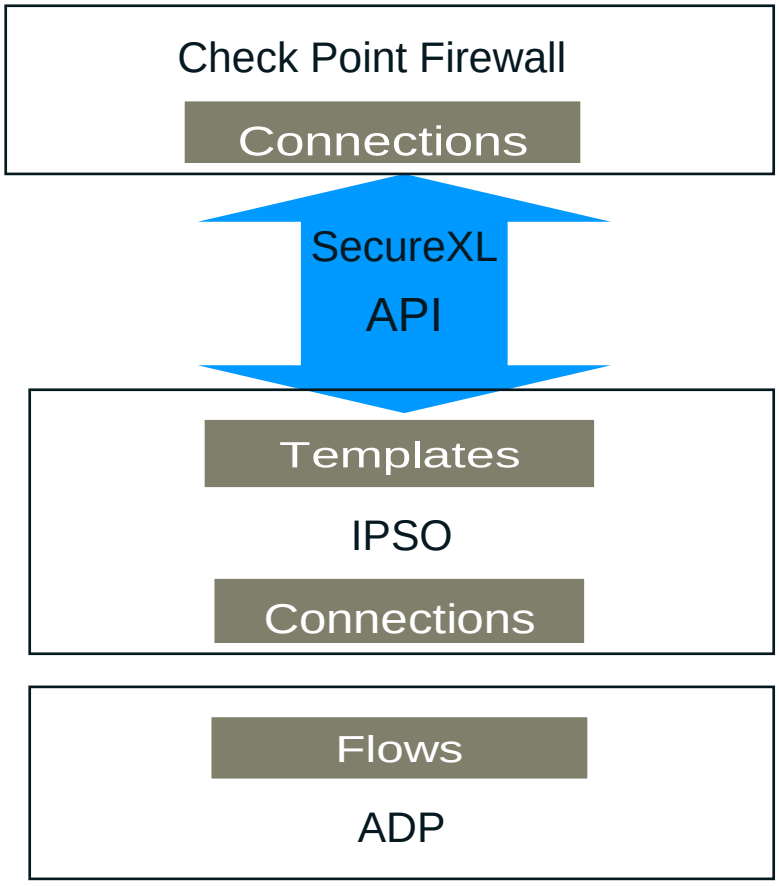
If the new packet does not match an existing connection template, then a round trip to the VPN-1 Power application is required to apply the security policy (rules). This path involves the greatest overhead.



10 The SecureXL Application Programming Interface (API)

The VPN-1 Power application communicates with Nokia IPSO software through the SecureXL API. The API supports the exchange of information between VPN-1 Power and IPSO relating to packets and packet streams, enabling IPSO to take over validation of subsequent traffic only initially validated by the VPN-1 Power application itself.

The type of information exchanged via the API includes initialization, configuration, flow, security association and connection handling, and statistics. The figure below illustrates the relationship between the VPN-1 Power application software and IPSO operating system software running on Intel x86 hardware and the network processor hardware.



11 The API History

The SecureXL 1.0 API introduced VPN and firewall throughput acceleration. Version 1.5 of the API improved connection rate by further delegating handling of the connection setup packets (except the very first one – the SYN packet) to IPSO. That means when the firewall offloads a connection, it supplies the connection's initial TCP state so IPSO can pick up from that state and continue handling the connection from that point on.

Version 2.0 increased connection rate by introducing the concept of connection templates by looking at four attributes for a match. The attributes are SrcAddr, SrcPort, Proto, DestAddr, and DestPort (the SrcPort is masked out). When the very first packet comes into the ingress port of the Nokia security appliance, it goes up to the firewall for processing since no connections have been validated. After the firewall processes and validates the packet, it stores the connections in the connection table, and offloads the connections to IPSO via SecureXL API. At the same time, the firewall also creates a template containing for attributes for a match through SecureXL API. The template stays within Nokia IPSO. When an inbound packet arrives and there is no match in the SecureXL connection table, a template match can now be attempted to avoid a round-trip to the firewall application for connection validation.

Version 2.1 further incrementally enhances connection rate and security by

- Allowing IPSO to delete connections without getting delete requests from the firewall (auto-expire)

- Allowing IPSO to handle certain short connections completely on its own (delayed notifications); this feature is particularly beneficial for systems processing many short-lived connections because syncing less often (as opposed to syncing on every connection) requires less CPU cycle.

- Allowing IPSO to implement an enhanced method of tracking the TCP state of TCP connections, including examining the TCP sequence and ACK numbers in addition to the TCP flags (TCP sequence validation)

TCP sequence validation by default is disabled in IPSO. To use this feature, users should enable it in both SmartDashboard and in IPSO using Voyager. The auto expiry and delayed notification features can also be enabled or disabled using Voyager. They are enabled by default. After enabling/disabling any of these features, policy needs to be pushed for the change to take effect.

Version 2.12 of the API supports acceleration of multicast traffic.

12 SecureXL 2.22 Features

SecureXL 2.22 adds VPN routing capabilities and enhances connectivity to support VPN in dynamic routing environment. SecureXL 2.22 is fully implemented in IPSO 4.2 and IPSO 6.0.

- VPN Link Selection - allows multiple external interfaces to be configured for tunneling the VPN packets. It allows the firewall to create, maintain, and update a table of Link Selection entries in IPSO. The firewall can specify which link needs to be used for a given SA. If that link goes down, the firewall can update IPSO to start using a different link for the same tunnel.

- Dynamic VPN Routing – allows the VPN domain to be determined dynamically instead of having to configure a static VPN domain. With Dynamic VPN routing enabled (using SmartDashboard) connections can transition from clear text to encrypted or from encrypted to clear text, based on the route taken by the connection. The connection properties adapt to the route changes between an external interface (untrusted) and an internal (trusted) interface by communicating in encrypted or clear text respectively.

- Wire Mode Connections – allows trusted traffic to pass through without stateful inspection. If an internal interface is configured as wired (trusted) and the VPN community is configured as wired, then the traffic passing through the internal interface and getting encrypted using the VPN community will skip any stateful inspection. This increases the connectivity at lower security for traffic between wired interfaces and a wired VPN community.

All VPN routing and connectivity enhancements mentioned above can be configured using SmartDashboard.

13 Software Requirements for SecureXL Software

SecureXL requires Nokia IPSO 3.8 (or later) and Check Point VPN-1/Firewall-1 NGAI R55 for IPSO 3.8 (or later).

14 Hardware

SecureXL is supported on all Nokia IP-series security appliance platforms that run the required Software (above). For applications involving large numbers of concurrent connections, additional memory may be advised because SecureXL consumes more memory per connection than Firewall Flows in earlier versions of IPSO. The increased memory usage reflects the additional information stored in the connection table relative to Firewall Flows. The tables below show the maximum number of concurrent connections of given RAM installations when running SecureXL for flash-based and disk-based systems.

Firewall Memory Settings for Disk-based IP Security Platforms

DRAM	Max Connections (standalone, VRRP)	Max Connections (2-node IP Cluster)	Max Connections with Web Intelligence	Hash table size	Memory pool size	Max memory pool size
256 MB	36,000	n/a	n/a	2 MB	48 MB	64 MB
512 MB	135,000	n/a	n/a	4 MB	196 MB	256 MB
1 GB	360,000	180,000	140,000	8 MB	400 MB	512 MB
2 GB	725,000	362,000	325,000	16 MB	800 MB	900 MB

Firewall Memory Settings for Flash-based IP Security Platforms

DRAM	Max Connections (standalone, VRRP)	Max Connections (2-node IP Cluster)	Hash table size	Memory pool size	Max memory pool size
512 MB	90,000	45,000	4 MB	128 MB	196 MB
1 GB	225,000	112,000	8 MB	256 MB	400 MB
2 GB	725,000	362,000	16 MB	800 MB	900 MB

CAUTION: Be aware that configuring the firewall for a higher connection capacity than the installed memory can support may result in erratic operation – disabling SecureXL, dropping packets, or system panic.

15 VPN Acceleration

SecureXL accelerates both site-to-site VPN and client VPN traffic when configured to use the encryption algorithm supported on the hardware accelerator card. To see the supported encryption algorithm, use 'fwaccel stat.'

Nokia cryptographic accelerators compatible with SecureXL are as follows:

Nokia Encrypt Accelerator Cards

Part Number	Platforms
Onboard	IP130, IP260, IP265, IP390, IP560, IP690, IP2450
NIF4530	IP290
NIF4214	IP350, IP355, IP380, IP385
NIF4427	IP560, IP690
NIF4400	IP1220, IP1260
NIF4420	IP1220, IP1260, IP2450
NIF4454	IP1280
NIF4503	IP2250
NIF4523	IP2250, IP2255
NIF4515	IP150

16 Configuring SecureXL

SecureXL is on by default when running IPSO 4.2. SecureXL is on by default when running IPSO 3.8 or later on the IP2250 security appliance. SecureXL must be enabled for the Accelerated Data Path (ADP) technology to function properly, and it should not be disabled. Disabling SecureXL on the IP Security platforms using ADP will severely degrade performance.

In earlier IPSO versions, SecureXL is off by default. You may enable, configure, and debug SecureXL using the capabilities of the fwaccel command or by using by cpconfig.

Command	Function
nokia[admin]# **fwaccel on**	Enables SecureXL
nokia[admin]# **fwaccel off**	Disable SecureXL
nokia[admin]# **fwaccel stat**	Checks overall SecureXL statistics
nokia[admin]# **fwaccel ver**	Shows SecureXL/FW version
nokia[admin]# **fwaccel cfg-configure**	Shows SecureXL parameters
nokia[admin]# **fwaccel stats-print**	Shows SecureXL statistics
nokia[admin]# **fwaccel conns-print**	Shows SecureXL's connection table
nokia[admin]# **fwaccel templates-print**	Shows SecureXL's templates table
nokia[admin]# **fwaccel dbg-set**	Sets debug flags
nokia[admin]# **fwaccel help**	Shows full details of the options

17 What is accelerated by SecureXL

The following protocols and environments are accelerated by SecureXL, subject to limitations described in the following section.

Throughput Acceleration:

- TCP, UDP, and traffic carried over those protocols
- IPSec VPN acceleration
- Mulitcast forwarding; PIM (from IPSO 3.9 for IP2250 & IP2255; from IPSO 4.2 for all platforms)
- GRE and ESP

Connection Rate Acceleration:

- Unencrypted TCP, UDP, and traffic carried over those protocols (when not using NAT)
    - Particularly effective on HTTP 1.1 traffic
    - Even more effective on HTTP 1.0 traffic (HTTP 1.0 uses a separate connection for each HTTP component)

18 SecureXL Limitations

SecureXL technology involves the VPN-1 Power application delegating certain decisions to software running on the other side of the SecureXL API in IPSO. By making these decisions closer to the network interface, at the hardware interrupt level or in the network processor, substantial overhead can be removed (queuing, copying, context switches, etc.). The types of decisions that can be made quickly are the ones that will benefit most from the reduced overhead. On the other hand, complex decisions that take a large amount of time relative to the overhead involved don't have as much opportunity for optimization.

The SecureXL API and IPSO support delegation of those decisions that benefit most from reduced overhead – involving the most common type of traffic and security policies. The evolution of SecureXL over time will address the most significant remaining limitations.

Certain types of traffic, and certain elements of the security policy, can negate the benefits of SecureXL, either for particular packets or for all traffic. For example, when setting up a new connection, in order to execute a rule where the source or destination is a domain, this decision (which is not delegated through the SecureXL API to IPSO) must be executed by the VPN-1 Power application itself. All new connections would need to be examined with this rule by the application itself rather than by IPSO, and so the use of this rule negates SecureXL's template connection rate acceleration for all traffic matching this rule and below.

To get the most out of SecureXL performance acceleration, the choice of Check Point applications and features, the rules that make up the security policy, and even potentially the ordering of the rules, should be carefully chosen based on the information below.

18.1 Incompatible Check Point Applications

- FloodGate-1 (automatically disables SecureXL and enables Firewall Flows)
- SmartView Monitor (traffic charts and contents don't account for packets that SecureXL device handles)

18.2 Traffic Limitations

The following traffic is not throughput nor connection-rate accelerated by SecureXL.

- Traffic types other than TCP, UDP, PIM, GRE, ESP
- First packets of any new TCP session, unless a "template" exists
- First packet in a UDP session
- Traffic matching certain Firewall rules
  - with a service that uses a resource

  - for dropping or rejecting traffic
  - where the rule's source or destination is the gateway itself
  - with a security server
  - with user or session authentication
- Directed broadcast traffic
- Traffic across an Access Control List – enabled interface
- IPv6 traffic
- VPN encryption algorithms that are not supported by the VPN accelerator card
- Traffic requiring any non-trivial processing such as fragmentation, IP compression, IP options
  - Exception: As of IPSO 4.2, SecureXL can manage QoS of traffic accelerated by SecureXL (but not ADP)

The following traffic is not connection-rate accelerated by SecureXL.

- Non-TCP/UDP connections such as PIM, GRE, ESP
- Protocols that are not connection intensive such as SMTP, FTP, RPC, NFS, NNTP, NTP
- Complex connections such as IPSec VPN, FTP, H.323, etc.
- Traffic in environment using NAT (for security, NAT addresses can change and can be shared)

18.3 Firewall Rule Limitations

Certain security policy rules and rule properties invoke extensive algorithms that are not replicated across the SecureXL API. SecureXL would not necessarily enable significant acceleration even if they were replicated because of their complexity relative to application overhead. For optimum performance, the security policy should be designed, where possible, avoiding these rules and rule properties.

The following rule properties present in the security policy will disable connection-rate acceleration for traffic matching that rule and all traffic below it. (Throughput acceleration is not inhibited by the presence of rules with these properties.)

- Rules where source or destination is a domain
- Rules with complex objects: Time, (Source) Port Range, Dynamic, Domain
- Rules using complex services:
  - A "Match" field, those with "Enable reply from any port"
  - RPC, DCOM, DCE-RPC services
  - User, client, or session authentication

- Rules for non-TCP/UDP/GRE/ESP connections
- Rules with SYN Defender or Small PMTU enabled
- Client or session authentication involved with the rule
- Rules where the service has an INSPECT handler (e.g. FTP control connection)
- Explicit VRRP rules (implicit rules are OK)

Note: Use 'fwaccel stat' to list which rules disable SecureXL or templates and move the those rules to the bottom of the rulebase. If most of the connections are NAT or VPN connections, then the rule-base order change is not necessary as no templates will be used. In such setting, make sure the most frequently used rules are placed on top to gain performance improvements.

Also, when installing a policy containing a restricted rule, console messages will appear to indicate that Connection Templates will not be created due to the rules that have been defined. This warning should be used to fine-tune policy to optimize performance.

18.4 SmartDefense Configurations that Disable Connection-rate Acceleration for All Traffic

- SYN Attack Configuration
- Network Quota
- Peer-to-Peer
- Instant Messenger
- SSH – Detect SSH over non-standard port
- Application Layer: Web Intelligence:
    - Block HTTP on non-standard port
    - Block HTTP Malicious Encodings

18.5 SmartDefense Configurations that Disable Throughput and Connection-Rate Acceleration For Relevant Traffic

- Spoofed Reset protection (forwards RST packets to the firewall)
- Block ASN.1
- Block WINS Replication attack
- Block WINS Name Violation
- DNS Protection
- SNMP Checks

- SUN-RPC Program Lookup
- SSH enforcement
- Routing Protocols Check (RIP, BGP, OSPF, IGMP)
- Content Protection:
    - MS-RPC
    - MS-SQL
- VPN Protocols:
    - PPTP enforcement
    - SSL enforcement
    - Block IKE Aggressive Exchange (disables IKE Acceleration for client to server direction only)
    - IKE enforcement (disables IKE for client to server direction only)

18.6 SmartDefense Configurations that Disable Throughput and Connection-Rate Acceleration for All HTTP Traffic

- Content Protection:
    - Malformed JPEG
    - Malformed ANI files
- Application Layer: Web Intelligence:
    - HTTP Header Spoofing Check
    - Directory Listing
    - Error Concealment
    - ASCII only response header

18.7 Configurations that Disable SecureXL for All Traffic

- Using ClusterXL with the “sticky decision” function
- Check Point FloodGate-1 (uses older Firewall Flows instead of SecureXL)
- Many Check Point SmartDefense configurations:
    - Sequence Verifier (unless you enable “Sequence Validation” in Voyager)
    - ISN Spoofing
    - TTL Check

- IP ID Check
- Application Intelligence Check:
    - POP3/IMAP Security
    - Mail Security Server
    - FTP Security Server
- Microsoft Networks – File and Print sharing
- Block NULL CIFS sessions
- Block Popup Messages