

Voyager Web Application Vulnerability Details

Prepared for:

Sanitized Release

Prepared by:

FishNet Security

Piranha Team

Security Services

Date Prepared:

October 8th-24th, 2003

FishNet Security
1710 Walnut
Kansas City, MO 64108

Phone: 816-421-6611
Toll Free: 888-732-9406
Fax: 816-421-6677

The information transmitted in this document is intended only for the addressee and may contain confidential and/or privileged material. Any interception, review, retransmission, dissemination or other use of or taking of any action upon this information by persons or entities other than the intended recipient is prohibited by law and may subject them to criminal or civil liability.

FishNet Security has conducted work associated with this report as an independent, third party vendor, and maintains no ownership or interest in the Client.

Copyright © 2003 FishNet Security. All rights reserved. The FishNet Security logo is a registered trademark of FishNet Security. All other products and company names mentioned herein are trademarks or registered trademarks of their respective owners.

General Information 1
 Company Background1

Corporate Profile..... 2
 FishNet Security Vulnerability Research Team2

Executive Summary..... 10
 Purpose.....10
 Perspective.....10
 Issues11

Findings and Recommendations 11
 Findings.....11
 Threat Vectors12
 Recommendations15

General Information

Company Background

Founded in 1996, FishNet Security has become one of the country's leading and most respected innovators in the network security industry. FishNet Security is focused exclusively on network security. Our roots are grounded in the engineering and technical aspects of network security as opposed to consulting firms that have ventured resources into the network security arena. Our business foundations offer strength and stability that set us apart from the "dot-com" model.

Commitment to our Customers

Headquartered in Kansas City, Missouri, FishNet Security is committed to being the largest network security company in the Midwest. In order to provide superior customer service, FishNet has regional offices in St. Louis, Dallas, Minneapolis, and New York. Our management team works to ensure high level of service through frequent and direct contact with our customers.

Engineering Expertise

FishNet Security offers a technical staff with experience, training and industry certifications such as Check Point Certified Security Expert, Cisco Certified Internetworking Engineer (CCIE), Certified Information System Security Professional, Microsoft Certified System Engineer and more. Our engineers are certified in industry leading security product lines, and in the networking, operating system and routing foundations that underscore successful implementations.

Web Application, Web Service, and Application Server Assessment Tools

FishNet Security utilizes a variety of proprietary, public source and commercial software tools to perform assessment services. The choice of tools is based on the Client technical environment and desired scope of work. Not all tools listed may be utilized for this project.

SPI Dynamics [WebInspect](#) and a variety of other tools that can include Sanctum's [AppScan](#), [Nstalker CGI Scanner](#), [SandSprite Sleuth](#), [Form Scalpel](#), and open source tools like [SPIKE Proxy](#) are utilized for comprehensive web application, service, and protocol level assessments. SPI Dynamics [WebInspect](#) and Sanctum's [AppScan](#) are licensed products, which FishNet Security is authorized to resell. The other tools used comprise a mix of licensed and open source tools.

Corporate Profile

FishNet Security Vulnerability Research Team

The FishNet Security Vulnerability Research Team is comprised of highly technical employees and is designed to be flexible to meet the varying needs of FishNet and our clients. The following individuals participated in the vulnerability research on this project:

Arian J. Evans – Initial vulnerability discovery and threat analysis

Senior Security Engineer

Office: 816-421-6611

Email: arian.evans@fishnetsecurity.com

Trey Keifer – In depth vulnerability analysis and creation of various PoC (Proof of Concept Code)

Security Engineer – Level II

Office: 816-421-6611

Email: trey.keifer@fishnetsecurity.com

Brandy Peterson – IPSO and Voyager expertise, Voyager Best Practices documentation

Director of Technology

Office: 816-421-6611

Toll Free: 888-732-9406

Email: brandy.peterson@fishnetsecurity.com

Executive Summary

Purpose

FishNet Security has performed a partial application security assessment on the Nokia Network Voyager Interface, along with full vulnerability research regarding discovered vulnerabilities. An application security assessment is the process of identifying security risks and baselining the security posture of a specific application. A full assessment may take weeks or months of man-hours depending on the complexity of the application.

Due to time and complexity of assessing an entire platform, only a partial assessment has been performed on the Nokia Network Voyager application. FishNet Security's assessment team does offer comprehensive evaluation identifying threats, vulnerabilities, and suggesting solutions to specific security issues. In addition, FishNet Security's assessment services team performs vulnerability research on previously unknown vulnerabilities discovered during the course of assessment activities. With this information, an organization can design and implement a strategy to reduce overall risk exposure to its application(s).

Nokia did *not* contract FishNet Security to conduct an independent application security assessment of the web based Network Voyager application. The vulnerability research contained in this report was stimulated by and is the byproduct of other unrelated vulnerability research.

FishNet Security follows the Full Disclosure Policy v2.0 responsible reporting guidelines which may be found at <http://www.wiretrip.net/rfp/policy.html>.

Perspective

All exploitive testing of the Nokia Network Voyager platform was done blind, e.g.—as an unauthenticated, unauthorized user of the platform attempting to subvert the platform or gain administrative access.

Confirmation of exploitation, and further research into various commands that could be injected, were performed as a fully authenticated, administratively authorized user. This access is, however, in no way needed to inject hostile script into the Network Voyager application.

Execution of script injected into the application is predicated upon an authorized user being authenticated to the application, and viewing the appropriate logs by which scripts may then be transparently executed.

The application assessment testing was conducted by FishNet Security in Kansas City, MO.

Issues

Remote Root access and configuration abuses possible through script/command injection into Nokia's Network Voyager web administrative interface.

"Nokia Network Voyager is an SSL-secured, web-based element management interface to Nokia IP Security Platforms. Enabled via the Nokia IPSO operating system (OS), Network Voyager is used to configure and monitor individual Nokia IP Security Platforms. Through the simple, yet powerful user interface of Network Voyager, users can point any web browser at an individual Nokia IP Security Platform and immediately manage the device."

--Nokia Website

Default HTTP access issue: Nokia Network Voyager is *not* an SSL-secured management interface to Nokia IP Security Platforms by default. By default, Nokia Network Voyager is a clear-text enabled management interface: HTTP. Wrapping the platform's HTTP communications in an SSL tunnel is entirely optional, not enabled by default, and in no way needed to manage the platform. The same can be said for the use of telnet versus SSH for console access.

Use of SSL for the Nokia Network Voyager interface, and restriction of allowed clients to the Nokia Network Voyager interface are both vendor-recommended configurations. However the default protocol is HTTP and there is the potential that, out in the real world, HTTP will be left enabled in an unrestricted manner (we have seen this many times in assessment testing). This leaves the Nokia IP Security Platform in a very insecure state.

This insecurity could be the result of several issues. At the core, shipping the platforms HTTP enabled with open access by default means that many devices will be deployed exactly the way they are shipped. In the case where Voyager deployments have been properly secured by SSL and client access restrictions, these security measures are often later removed during trouble shooting, moving or upgrading systems, personnel changes, or swapping out a failed Nokia appliance for a replacement (all factory default) appliance.

Script Injection Issues: Through testing independent and unrelated to the Nokia Network Voyager product, FishNet Security's assessment team discovered unusual behaviors in the Voyager application creating the belief that deliberate subversion of the application might be possible. After careful scrutiny and testing, these behaviors led to remote root access via script (command) injection affecting IPSO versions 3.5, 3.6 and 3.7, and possibly earlier (untested) versions as well.

The crux of the script injection issue centers on the behavior of the httpd access log interface. Because this access log code does not properly handle or sanitize data that is returned to the client, it permits a malicious user to submit any arbitrary request to the httpd listener which will be logged exactly as submitted. If that request happens to contain malicious code, upon viewing the httpd access log in Network Voyager by a valid user's, their web browser will execute that malicious code under their privilege level.

Detailed threat vectors and a summary of attack trees are documented under findings, but the high-level summary of threat vectors is:

1. Malicious entity gains access to the httpd interface, via direct access, IP spoofing, remote client control, etc.
2. Administrator must access the Network Voyager httpd log viewer after script injection, whether through normal trouble shooting or review, or prompted through social engineering, etc.

Specific actions we were able to take via script injection into the httpd logs include:

- remote creation of new administrative account
- automatic setting of new account password (enabling account)
- disabling firewall daemon and ACLs
- rebooting Nokia IPSO appliance
- yield a wide-open box, no firewall, new remote admin user, full remote control

Findings and Recommendations

Findings

The Nokia Network Voyager access log web interface (<http://nokiaappliance/cgi-bin/httpdaccesslog.tcl>) vulnerable to Script Injection which allows an unauthenticated user the ability to create admin level users, reboot the device, and essentially perform any other administrative-level functions through the use of malicious JavaScript.

By submitting raw (non-URL encoded) http requests to the device an unauthenticated user is able to insert malicious JavaScript directly into the voyager access log. When this access log is viewed by an authenticated administrator via the web interface the malicious code is executed by the client web browser and a variety of functions can be performed on the device.

The easiest way to submit a request to the Voyager interface is by using a local web proxy. While a telnet client or even netcat could be used, negotiating syntax and the potential need for an external SSL wrapper makes this tedious. For the Windows platform, Achilles and Spike Proxy are both free tools that could be utilized for this, and automate SSL negotiation if SSL is in use. We use SPI Dynamics SPIProxy, a commercial tool (part of WebInspect) and Spike Proxy on Windows. For Linux or BSD, there are a variety of open source tools that can be used, from HTTPPush, to the SPIKE toolkit, to Ettercap, to Firebird browser extensions, etc.

The important thing is that you submit a raw request to the HTTP or HTTPS interface that gets logged. As for the script injected, you can use your imagination and basically go through the Voyager interface and TCL scripts and postulate what all forms can be automatically posted, scripted actions taken, etc. Virtually anything can be done with a script that the user can perform via a browser inside the Nokia Network Voyager interface.

Other tools that may be utilized to exploit various threat vectors are packet crafters and packet injectors, TCP session ID predictors, and various man in the middle utilities to inject commands into existing TCP sessions.

After the malicious code is successfully injected into the logs, the httpd access logs will then need to be viewed by a user with administrative privilege to execute the code. There are a number of conditions under which an administrator would normally review the httpd access logs, but in addition, a malicious attacker will likely attempt various forms of either service interruption (e.g. –DDoS) or social engineering to stimulate the administrator to review the access logs. Code could also be injected into the logs to alert the malicious attacker that the logs had been viewed. A simple HTTP GET to the malicious attacker's web server would leave an entry in that server's web logs that the malicious attacker could monitor for. A tool like SWATCH could be used to automatically monitor these logs for entries from poisoned Nokia IP Security Platforms, to automatically alert the malicious attacker when his scripts had been executed and a new system was vulnerable to exploitation. In this way notification of successful system compromise could be automated.

Sample attack scenario, step-by-step:

1. Malicious entity identifies Nokia IP Security platform belonging to Company X and identifies that the Nokia Network Voyager interface is listening on 172.16.1.1 port 443 with no client access restrictions.
2. Malicious attacker injects script into the device including script to create a new administrative user, set the password, and do an HTTP GET to a third, external web server with a unique, identifiable GET string.
3. Malicious attacker calls Company X, identifies himself as Nokia support, and asks for the firewall administrator. Attacker explains Nokia is calling customers to warn them of the new xipocsic threat to the platform, and encourages them to review their httpd access logs as soon as possible (in addition to other logs, of course).
4. Company X Firewall administrator reviews Nokia Network Voyager httpd access logs, finds nothing resembling the xipocsic threat, and closes their browser. Unbeknownst to the administrator, depending on the scripting ability of the malicious attacker, several pop-under boxes have executed script and closed creating another administrative user, setting the password and enabling the user, and doing an HTTP GET to an external web server.
5. Malicious attacker has SWATCH monitoring the logs of the external web server used for this hacking, and automatically kick off an alert via email when it notices a certain type of HTTP GET in the logs. This alert is all the attacker needs to know they have *another* compromised platform waiting for them to exploit.

If SSL were enabled, or interface access restrictions enabled, there would be many more steps to injecting the code, including predicting TCP session ISN, SSL Session ID, and blindly spoofing source IP until a valid source IP (allowed to connect) could be located and used for injection. For a remote attacker, using forged packets would mean this was all done quite blind, and almost impossible to tell when successful. For an attacker local to the network allowed to connect (or a closely adjoining network) it would be much easier to exploit this. In addition to being able to inject packets into valid TCP sessions, they could spoof IPs from a valid source and also potentially capture the responses to the spoofed packets, meaning the work wouldn't be blind at all...

Threat Vectors

To fully understand the various ways script may be injected into the Nokia Network Voyager interface, what the various risk levels are to those attacks, and how they may be mitigated or prevented, we outline below a summary of the attack trees created during testing to understand the varying potential risks. In terminology, we refer to 'unrestricted' as an interface anyone can connect to, and 'restricted' as an interface only specific IPs can connect to, though we realize some people prefer to restrict by network. As for risk level, HIGH refers to a trivial attack anyone could execute, MEDIUM refers to an attack only a skilled and dedicated hacker would attempt, and LOW refers to something likely only a researcher would take the time and effort to attempt, or would have to be combined with

other issues and vulnerabilities, or other compromised hosts (as in the case of a remote attacker gaining local network access prior to launching attacks against Network Voyager).

Attack against HTTP interface, unrestricted, remote attacker:

- Direct TCP connection, requirements: trivial, risk: HIGH
- Blind Packet Injection, requirements: predicted TCP ISN, risk: MEDIUM
- TCP session injection, requirements: access to local network, risk: LOW

Attack against HTTP interface, unrestricted, local attacker:

- Direct TCP connection, requirements: trivial, risk: HIGH
- Blind Packet Injection, requirements: predicted TCP ISN, risk: HIGH (easier to predict ISN when harvesting connections locally in bulk)
- TCP session injection, requirements: access to local network, risk: HIGH

Attack against HTTP interface, restricted, remote attacker:

- Direct TCP connection, requirements: trivial, risk: LOW
- Blind Packet Injection, requirements: predicted TCP ISN, risk: MEDIUM
- TCP session injection, requirements: access to local network, risk: LOW

Attack against HTTP interface, restricted, local attacker:

- Direct TCP connection, requirements: trivial, risk: LOW
- Blind Packet Injection, requirements: predicted TCP ISN, risk: HIGH (easier to predict ISN by harvesting connections locally in bulk)
- TCP session injection, requirements: access to local network, risk: HIGH

Attack against HTTPS (SSL) interface, unrestricted, remote attacker:

- Direct TCP connection, requirements: trivial, risk: HIGH
- Blind Packet Injection, requirements: predicted TCP ISN, risk: LOW (very difficult to predict both ISN and negotiate SSL tunnel in the blind without use of third host for return packets)
- TCP session injection, requirements: access to local network, risk: LOW

Attack against HTTPS (SSL) interface, unrestricted, local attacker:

- Direct TCP connection, requirements: trivial, risk: HIGH
- Blind Packet Injection, requirements: predicted TCP ISN, risk: Medium (easier to capture and monitor response traffic to get ISN and negotiate SSL tunnel/session ID from local network)
- TCP session injection, requirements: access to local network, risk: HIGH

Attack against HTTPS (SSL) interface, restricted, remote attacker:

- Direct TCP connection, requirements: trivial, risk: LOW
- Blind Packet Injection, requirements: predicted TCP ISN, risk: LOW
- TCP session injection, requirements: access to local network, risk: LOW

Attack against HTTPS (SSL) interface, restricted, local attacker:

- Direct TCP connection, requirements: trivial, risk: LOW
- Blind Packet Injection, requirements: predicted TCP ISN, risk: Medium (easier to capture and monitor response traffic to get ISN and negotiate SSL tunnel/session ID from local network)
- TCP session injection, requirements: access to local network, risk: HIGH

The most important point to note is that if an attacker is local to the network, bypassing layer 3 (IP restrictions) or layer 4 (SSL) controls becomes much easier given layer 2 access. In addition, there is also the ability to spoof a valid IP to access a restricted interface, and monitor the response traffic, which a remote attacker would most likely be unable to do.

If more detail is needed regarding the distinctions between risk on local and remote attacks versus SSL and IP restrictions, please contact FishNet Security's CSIRT Team and we will be happy to go over the technical requirements to exploit these different scenarios in further detail.

Recommendations

Workaround: As a simple workaround, users may disable httpd access logging and thus defeat any attempt to inject script into the access logs.

Secure Nokia Network Voyager Deployment: Please refer to our best practices document for securing Nokia Network Voyager:

<http://www.fishnetsecurity.com/CSIRT/disclosure/Nokia/Securing.Nokia.Network.Voyager.pdf>