[Date]

From NXLog to log managers

By CyberApprentice

Introduction

NXLog is an open source log collector used to forward logs to a central repository. It is a lightweight agent that can format data through secure protocols across multiple platforms (e.g. Windows and \*nix). This writing will utilize NXLog community edition as an ingest tool to pipe logs to a central log server across Linux and Windows and utilize the available capabilities to better understand the power enabled to ease collection and monitoring of the plethora of data generated by applications throughout the systems.

Details on NXLog Community Edition

NXLog may be downloaded at their official website, https://nxlog.co. There are several products available: *NXLog Community Edition* – an open source alternative that provides the core functions of collecting and forwarding logs; *NXLog Enterprise Edition* – an enhanced paid version for the community edition provides features such as: support, agent management, additional modules, and greater reliability; *NXLog Manager* – a paid, central admin station to monitor and manage NXLog agents; and *NXLog Extras* – purpose built addons that collect logs from popular cloud services such as Microsoft Azure, Salesforce, and more. No matter the product, each offer intends to solve the requirements of “filtration, classification, correlation, forwarding, and storage” and generated event logs.[[1]](#endnote-1) In particular, it features Windows support when traditional logging applications were tailored towards \*nix systems or custom, in-house scripts, multithreading, formatting, prioritization, flow control, scheduling, filtering, altering and protocol selection, including secure and unsecure protocols. It is important to realize that NXLog is **not** a SIEM, and therefore does not provide a GUI, vulnerability detection, compliance checking, or automated analysis.

Log messages generated by applications and processes include information pertaining to verbose output iterated steps, status, errors, and debug messages, which are all classified as *events*. NXLog processes these events through unique modules. For example, UDP events are processed through *im\_udp* and *om\_udp* modules, which stand for “input module user datagram protocol” and “output module user datagram protocol”, respectively. NXLog processes logs through predefined filters. When the agent receives a log, it stores it in its raw format, adds default fields, and then the input module parses the raw format and derives additional fields for classification purposes. For example, if NXLog is passed a syslog message, it will extract data points such as the timestamp and IP address through predefined regular expressions within the syslog module. NXLog provides the ability to create additional modules and the user is able to append their own desired metadata extraction methods. Events can be broken into two types, structured and unstructured data. The methodology above is utilized to handle common unstructed data with common patterns such as syslog messages. Structured data is already formatted and standardized, allowing only for NXLog to reference the design of the datagram, parse it accordingly, and display key data points.

When it comes to forwarding logs over the network, NXLog provides flexibility and leaves the trade-off to the user. Modules available to send traffic include UDP, TCP, SSL/TLS, HTTP, HTTPS, and personal sockets. There is also the option to store events in files, databases and executables. An important that must be highlighted is the difference between the transport protocol, UDP or TCP. Popular protocols like syslog are sent over UDP because logs do not require the overhead as TCP does. UDP is a “fire & forget” protocol where no session is built up. TCP establishes this session through a handshake process and including sequence numbers and acknowledgements to resend any lost data. As the name implies for UDP, if it doesn’t make it, then it is lost or up to the programmer to assure important data is not lost. This the main difference equating to best effort (TCP) vs. not guaranteed (UDP) transport layer protocols. To further handle the bandwidth, there are additional options to reduce overall bandwidth. Although individually, logs are very small in size (<1MB), having numerous applications and services running across multiple machines consistently transmitting log data on the wire can soak up and limit available bandwidth. In the documentation, it is suggested to filter and deduplicate events, optimizing the modules by removing fields that are not of concern, and compressing the packets.[[2]](#endnote-2) In addition, encryption is available through the SSL and TLS certificates. The certificates are generated by the user and the recommended key size is 2048-bit size.

As stated previously, NXLog provides multiplatform support: Windows, Solaris, MacOS, Linux, Open & Free BSD, and IBM AIX. It is also able to be integrated with various different technologies and applications such as Dell iDRAC, Cisco appliances, Amazon Web Services, Snort, VMware vCenter, Docker, and much more. In our case, we will test with Windows Event Viewer, Cisco Nexus 4500 (syslog), Graylog, VMware vCenter, Dell iDRAC and Microsoft Azure. The listed items were chosen because we are using the community edition and it is what is available for testing. However, they cover a wide variety of different type of sources that logs could be pulled from and centralized.

Deploying NXLog Community Edition

Installation:

1. Download at <https://nxlog.co/products/nxlog-community-edition/download>
   1. This guide will use ubuntu 18
2. Sudo dpkg -i nxlog-ce\_2.10.2150\_ubuntu\_bionic\_amd64.deb
   1. will fail because of dependencies
3. sudo apt -f install
   1. download dependencies that caused the failure
4. Sudo dpkg -i nxlog-ce\_2.10.2150\_ubuntu\_bionic\_amd64.deb

Configuration:

1. Mv /etc/nxlog/nxlog.conf /etc/nxlog/nxlog.conf.orig
2. Sudo vi /etc/nxlog/nxlog.conf
3. Append the following:
4. Allow nxlog permission to access the files above
5. sudo setfacl -m g:nxlog:rx /var/log/messages

https://www.loggly.com/ultimate-guide/centralizing-windows-logs/

1. https://nxlog.co/documentation/nxlog-user-guide/about-nxlog.html [↑](#endnote-ref-1)
2. https://nxlog.co/documentation/nxlog-user-guide/reducing-bandwidth.html [↑](#endnote-ref-2)