[Skip to main content](https://lms.alnafi.com/xblock/block-v1:alnafi+DCCS102+2025_DCCS+type@vertical+block@92a4a6617c534bffa6a37d05152b7087?exam_access=&recheck_access=1&show_bookmark=0&show_title=0&view=student_view#main)

Articles to read. Translate them into your native language using google and read them. Do not worry as you will learn all these things on the Ghanimah Platform.

Reference <https://nxlog.co/whitepapers/dns-logging>

Attackers are using DNS for data theft, denial-of-service, and other malicious activity. Proactive monitoring of DNS activity can help network administrators quickly detect and respond to these threats.

**About the Domain Name System (DNS)**

The Domain Name System (DNS) provides a hierarchy of names for computers and services on the Internet or other networks. Its most noteworthy function is the translation of domain names such as *example.com* into IP addresses. DNS is required for the Internet to function, operates on a global scale, and is massively distributed.

DNS servers normally accept messages on UDP port 53. The DNS protocol has two message types, *queries* and *replies*; both use the same format. These messages are used to transfer *resource records* (RRs). A RR contains a name, a time-to-live (TTL), a class (normally *IN*), a type, and a value. For example, an *A*-type resource record specifies the IPv4 address associated with a domain. The domain name space is divided into DNS *zones* and a server is considered *authoritative* if it has authority for a particular zone.

An "A" resource record for example.com

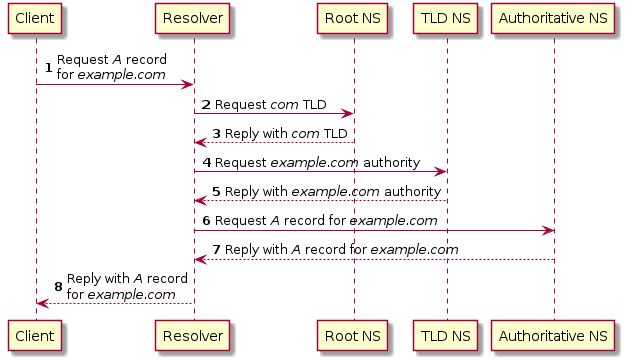
example.com. 3600 IN A 93.184.216.34

A DNS request involves the following parts (in the common case):

|  |  |
| --- | --- |
| Recursing resolver | The *resolver* receives the request from the client and makes further requests as necessary to obtain the authoritative record. |
| Root nameserver | A *root* nameserver can be queried to acquire more information about a particular top-level domain (TLD), such as *com*. |
| TLD nameserver | A *TLD* nameserver provides information about domains ending in a particular TLD. |
| Authoritative nameserver | Finally, an *authoritative* nameserver is one that contains the actual DNS records for a particular DNS zone. |

Example 1. DNS request and resolution

For example, consider a user attempting to connect to a website at*example.com*. Before attempting to connect, the system must acquire the IP address for that domain (an *A* record for IPv4). Here is the basic process for resolving an uncached record:



**DNS security concerns**

When the Domain Name System was designed, security was not a major consideration. Now, malicious actors are using DNS for data theft, denial-of-service attacks, command-and-control, and other malicious activity. According to an EfficientIP survey for 2018[[1](https://nxlog.co/whitepapers/dns-logging#_footnotedef_1)], the average cost of a DNS attack was $715,000 (57% higher than in 2017) and 77% of organizations were subject to DNS attack.

Common types of DNS attacks include:

**DNS hijacking**

An attacker can perform DNS hijacking by manipulating either user workstations or DNS servers. In the first case, malware is used to modify a workstation’s configured name servers, causing DNS requests to be sent to malicious servers instead. Alternatively, a DNS server may be compromised and configured to return incorrect replies. Either way, users are directed to an attacker’s site, allowing the attacker to steal user credentials (phishing), generate traffic (pharming), distribute malware, or publish a defaced version of the website.

In early 2019, FireEye identified a DNS hijacking campaign[[2](https://nxlog.co/whitepapers/dns-logging#_footnotedef_2)] of "unprecedented scale" that has had a "high degree of success" targeting victims across the globe. Soon after, the United States Cybersecurity and Infrastructure Security Agency released[Emergency Directive 19-01](https://cyber.dhs.gov/ed/19-01/), which outlines actions for mitigating the risk of DNS hijacking—including DNS auditing and monitoring. The recent Sea Turtle[[3](https://nxlog.co/whitepapers/dns-logging#_footnotedef_3)] campaign is a state-sponsored DNS hijacking attack that has targeted at least 40 different organizations across 13 different countries, using man-in-the-middle attacks to harvest user credentials.

**DNS tunneling**

DNS queries and responses can contain data payloads capable of transporting malware, stolen data, command-and-control information, or bidirectional protocols such as SSH. DNS traffic is often considered benign and not monitored. Therefore, DNS tunneling may be of particular interest to an attacker as a covert communication channel.

Many types of client malware make use of DNS tunneling, including point of sale (PoS) malware like MULTIGRAIN[[4](https://nxlog.co/whitepapers/dns-logging#_footnotedef_4)], remote backdoors such as DNSMessenger[[5](https://nxlog.co/whitepapers/dns-logging#_footnotedef_5)] and DNSpionage[[6](https://nxlog.co/whitepapers/dns-logging#_footnotedef_6)], and botnets like JAKU[[7](https://nxlog.co/whitepapers/dns-logging#_footnotedef_7)]. There are also several open-source DNS tunnel implementation examples: [dnscat2](https://github.com/iagox86/dnscat2),[iodine](https://code.kryo.se/iodine/), and[Heyoka](http://heyoka.sourceforge.net/).

**Various denial-of-service (DoS) attacks**

There are several different types of denial-of-service attacks that are used against DNS servers, including*NXDOMAIN*, *phantom domain*, and *domain lock-up* attacks. In each case, the attacker’s goal is to increase load on the server to the point where it is unable to answer legitimate requests.

One of the largest of these attacks was the 2016 Dyn distributed DoS (botnet) attack[[8](https://nxlog.co/whitepapers/dns-logging#_footnotedef_8)], which brought down many major sites in the United States and Europe.

**DNS cache poisoning**

Cache poisoning (or spoofing) occurs when a DNS resolver accepts an invalid resource record due to a vulnerability. An attacker may use a long time-to-live (TTL) value, during which time the data is retained in the resolver’s cache and the resolver is considered "poisoned". The result is similar to DNS hijacking. DNS cache poisoning is mitigated by cryptographic signatures such as those implemented by the Domain Name System Security Extensions (DNSSEC), but DNSSEC is not yet universally deployed.

By proactively monitoring DNS audit logs and query traffic, IT personnel can more quickly identify and respond to a DNS attack, reducing its impact.

**DNS data that can be collected and monitored**

There are many different DNS-related events that can be collected from a DNS server. The actual data available is dependent on the DNS server in use. Events that may be available for collection include, but are not limited to:

|  |  |
| --- | --- |
| Dynamic updates | If a DNS server is configured to allow DNS updates via Dynamic DNS (DDNS), these events should be available for auditing, including approval and denial of update requests. |
| Zone transfers | DNS databases can be replicated between DNS servers with zone transfers (the AXFR query type). These transfers, inbound and outbound, should be logged. |
| Rate limiting | A DNS server may implement rate limiting to help mitigate various types of DNS attacks, including response rate limiting (RRL) and client rate limiting. When rate limiting takes effect, some queries are ignored (for RRL) or client requests cannot be resolved. |
| DNS signing | Events may be logged for DNS transaction signing protocols including DNSSEC and TSIG. |
| Query logs | DNS servers often provide some form of query logging, also referred to as *analytical* logging. These events detail all requests that are handled by the server. |
| Resolution queries | Events may also be available for recursive lookups performed in order to resolve client queries. These logs may provide destination IP addresses and other details associated with these queries. These also could be categorized with *analytical* logs and may be included as part of query logging. |

This metadata may be available for query logging (analytical) events:

**Client IP address**

Source addresses can help administrators identify devices that may be compromised on a local network, or malicious actors on the public Internet.

**Domain name queried**

The domain name in requests can be compared with a list of known malware domains. Esoteric domain names or repeated lookups may indicate the presence of malware.

**Record requested**

The types of records requested may be an indicator of malicious activity. For example, the *TXT* record in particular is often used for command-and-control or DNS tunneling.

**Request flags**

There are several status flags associated with a DNS message, including whether the message is a request or response, whether the query is recursive, DNSSEC status, etc.

**BIND 9 name server logging**

There are two types of logging to consider for BIND 9 servers:

**BIND 9 logging**

The standard BIND 9 logging system uses *channels* and*categories*. Each channel logs messages to a specified Syslog facility or file. All log messages are organized into categories, which range from security related (such as *update-security*) to analytical (such as*queries*). Each category specifies one or more channels to use for log messages in that category. For more information and a full list of categories, see the [BIND 9 Administrator Reference Manual](https://www.bind9.net/manuals).

BIND 9 query log sample

01-May-2019 00:27:48.084 queries: info: client @0x7f82bc11d4e0 10.80.0.1#53995 (google.com): query: google.com IN A +E(0) (10.80.1.88)

**Configuration monitoring**

File integrity monitoring can also be implemented for the BIND 9 configuration files (for example, in the /etc/bind/ directory). This could be done as part of a system-wide DNS server auditing plan.

**Windows DNS Server logging**

There are four types of logging available for Windows DNS Server events.

**Analytical logging**

DNS analytical logging uses the Event Tracing for Windows (ETW) system to provide high-performance logging of all DNS transactions. The logs can be collected from theMicrosoft-Windows-DNSServer provider. This functionality is available beginning with Windows Server 2016, or as a hotfix for Server 2012 R2, and is the preferred method for collecting DNS Server transaction logs. DNS Server performance should not be affected except during very high query rates.  For more information, see[DNS Logging and Diagnostics](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/dn800669(v=ws.11)) on Microsoft Docs. For the Server 2012 R2 hotfix 2956577, see[Update adds query logging and change auditing to Windows DNS servers](https://support.microsoft.com/en-us/help/2956577/update-adds-query-logging-and-change-auditing-to-windows-dns-servers) on Microsoft Support.

**Native DNS auditing**

DNS auditing was also introduced alongside the analytical DNS logging changes—with Windows Server 2016, or 2012 R2 with hotfix 2956577—and is enabled by default. These audit logs are written to the Microsoft-Windows-DNS-Server/Audit log in the Windows EventLog. This is the preferred method for collecting DNS audit logs.

**Debug logging**

For Windows DNS Server versions prior to Windows Server 2012 R2, or on 2012 R2 without hotfix 2956577, "debug logging" can be enabled to record DNS queries and replies to a log file. This type of logging can affect DNS Server performance and is not intended to be enabled permanently. However, without support for analytical logging, DNS debug logging is the only way to collect DNS transaction information from Windows DNS Server. For more information about debug logging, see[Using server debug logging options](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2003/cc776361(v=ws.10)) on Microsoft Docs.

DNS Server debug log sample

4/21/2017 7:52:03 AM 06B0 PACKET 00000000028657F0 UDP Snd 10.2.0.1 6590 R Q [8081 DR NOERROR] A (7)example(3)com(0)

**Active Directory auditing**

On systems without native DNS auditing (prior to 2012 R2, or 2012 R2 without hotfix 2956577), DNS changes can be audited by enabling AD Directory Services auditing. These events are written to theMicrosoft-Windows-Security-Auditing log in the Windows EventLog. For more information, see the[AD DS Auditing Step-by-Step Guide](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc731607(v=ws.10)) on Microsoft Docs and[Who Moved the DNS Cheese? Auditing for AD-Integrated DNS Zone and Record Deletions](https://techcommunity.microsoft.com/t5/Core-Infrastructure-and-Security/Who-Moved-the-DNS-Cheese-Auditing-for-AD-Integrated-DNS-Zone-and/ba-p/256653) on Microsoft Tech Community.

**How NXLog Enterprise Edition can help with DNS monitoring**

NXLog Enterprise Edition provides several unique features for collecting and processing DNS logs:

**Native ETW collection**

NXLog includes an[im\_etw](https://nxlog.co/documentation/nxlog-user-guide/im_etw.html) input module that implements an ETW consumer. This allows NXLog to collect DNS analytical logs directly from ETW, without requiring events to be first written to disk.

**Parsing of DNS debug logs**

The[xm\_msdns](https://nxlog.co/documentation/nxlog-user-guide/xm_msdns.html) module offers built-in parsing of log messages read from Windows DNS Server debug logging files.

**Multiple types of Windows EventLog collection**

NXLog can be configured to collect Windows EventLog data as a local agent or remotely. NXLog can also use Windows Event Forwarding to collect EventLog data remotely while running on Linux. As a result, there are a variety of options for collecting DNS Server audit logs or Active Directory audit logs from a Windows DNS Server. See the[Windows EventLog](https://nxlog.co/documentation/nxlog-user-guide/windows-eventlog.html) chapter in the NXLog User Guide for more details.

**File integrity monitoring**

NXLog can be used to implement several types of checksum-based and real-time file integrity monitoring. This can be used to monitor BIND 9 configuration files or any other system files that are related to DNS. For more information, see[File Integrity Monitoring](https://nxlog.co/documentation/nxlog-user-guide/fim.html).

**Parsing and structured logging**

NXLog provides many tools for parsing event messages, and is designed around the concept of structured logging. Log data that is structured at the source can be retained as structured data, reducing the need for further processing. By handling and forwarding logs as structured data, events can be easily filtered, classified, and correlated as required for monitoring. For an introduction to NXLog’s handling of event data, see[Event Records and Fields](https://nxlog.co/documentation/nxlog-user-guide/events-fields.html).

**Output integrations**

Once log data has been collected and parsed, the structured data can be forwarded to another NXLog agent for centralized log collection or to a log analytics solution for monitoring and analysis. NXLog offers a range of extension and output modules capable of converting log data and forwarding it to other systems. NXLog supports a variety of SIEM[[9](https://nxlog.co/whitepapers/dns-logging#_footnotedef_9)] products and many MSSPs[[10](https://nxlog.co/whitepapers/dns-logging#_footnotedef_10)]. See[Integration](https://nxlog.co/documentation/nxlog-user-guide/integration.html) in the NXLog User Guide.

For more information about using these features to implement DNS logging for specific DNS servers, see the[DNS Monitoring](https://nxlog.co/documentation/nxlog-user-guide/dns-monitoring.html) chapter in the NXLog User Guide.

NXLog Ltd. develops multi-platform log collection tools that support many different log sources, formats, transports, and integrations. The tools help administrators collect, parse, and forward logs so they can more easily respond to security issues, investigate operational problems, and analyze event data. NXLog distributes the free and open source[NXLog Community Edition](https://nxlog.co/products/nxlog-community-edition) and offers additional features and support with the[NXLog Enterprise Edition](https://nxlog.co/products/nxlog-enterprise-edition).

This document is provided for informational purposes only and is subject to change without notice. Trademarks are the properties of their respective owners.

[1](https://nxlog.co/whitepapers/dns-logging#_footnoteref_1). [A New Era Of Network Attacks, 2018 EfficientIP Global DNS Threat Report](https://www.efficientip.com/resources/dns-security-survey-2018/), EfficientIP, retrieved 2019-05-14

[2](https://nxlog.co/whitepapers/dns-logging#_footnoteref_2). [Global DNS Hijacking Campaign: DNS Record Manipulation at Scale](https://www.fireeye.com/blog/threat-research/2019/01/global-dns-hijacking-campaign-dns-record-manipulation-at-scale.html), FireEye, Threat Research, retrieved 2019-05-14

[3](https://nxlog.co/whitepapers/dns-logging#_footnoteref_3). [DNS Hijacking Abuses Trust In Core Internet Service](https://blog.talosintelligence.com/2019/04/seaturtle.html), Cisco Talos Intelligence Group, Talos Blog, retrieved 2019-05-14

[4](https://nxlog.co/whitepapers/dns-logging#_footnoteref_4). [MULTIGRAIN – Point of Sale Attackers Make an Unhealthy Addition to the Pantry](https://www.fireeye.com/blog/threat-research/2016/04/multigrain_pointo.html), FireEye, Threat Research, retrieved 2019-05-14

[5](https://nxlog.co/whitepapers/dns-logging#_footnoteref_5). [Covert Channels and Poor Decisions: The Tale of DNSMessenger](https://blog.talosintelligence.com/2017/03/dnsmessenger.html), Cisco Talos Intelligence Group, Talos Blog, retrieved 2019-05-14

[6](https://nxlog.co/whitepapers/dns-logging#_footnoteref_6). [DNSpionage Campaign Targets Middle East](https://blog.talosintelligence.com/2018/11/dnspionage-campaign-targets-middle-east.html), Cisco Talos Intelligence Group, Talos Blog, retrieved 2019-05-14

[7](https://nxlog.co/whitepapers/dns-logging#_footnoteref_7). [JAKU, Analysis of a Botnet Campaign](https://www.forcepoint.com/sites/default/files/resources/files/report_jaku_analysis_of_botnet_campaign_en_0.pdf), ForcePoint Security Labs, retrieved 2019-05-14, page 32

[8](https://nxlog.co/whitepapers/dns-logging#_footnoteref_8). [DDoS attack that disrupted internet was largest of its kind in history, experts say](https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet), The Guardian, retrieved 2019-05-14

[9](https://nxlog.co/whitepapers/dns-logging#_footnoteref_9). SIEM: Security information and event management

[10](https://nxlog.co/whitepapers/dns-logging#_footnoteref_10). MSSP: Managed security service provider

Ref: <https://nxlog.co/dns-log-collection-on-windows>

[**DNS Log Collection - Part 2**](https://nxlog.co/dns-log-collection-on-windows)

  By adm     [0 Comments](https://nxlog.co/dns-log-collection-on-windows/#comments)

Be sure to read [Part 1](https://nxlog.co/dns-log-collection-and-parsing) and[Part 3](https://nxlog.co/dns-log-collection-on-linux) of our DNS Log Collection series, in case you missed them.

**DNS Log Collection on Windows**

If you need to reduce the cost of DNS security and increase efficiency through centralizing DNS log collection, where would you start? Answering this question requires knowledge and awareness of the challenges and opportunities available on the Windows platform. While[Windows DNS server](https://nxlog.co/documentation/nxlog-user-guide/windows-dns-server.html) is a common technology serving many types of organizations, from local domains to large multi-site enterprises, the possibilities are not necessarily that well-known within the context of comprehensive, site-wide log collection. This article distills the main concepts essential to planning and deploying such an implementation into this article, which serves as the second part of the DNS log collection series. To start, this article will touch on log sources that are generated by Windows DNS servers as well as the DNS requests of the clients they serve.

**Windows DNS Log Sources**

You may know that there are numerous ways of collecting DNS logs within the Windows environment:

* Collecting DNS query logs via[Sysmon](https://nxlog.co/documentation/nxlog-user-guide/windows-dns-server.html#sysmon_dns_query_logging)
* Collecting traces directly with [Event Tracing for Windows (ETW)](https://nxlog.co/documentation/nxlog-user-guide/windows-dns-server.html#dns_windows_etw) DNS Providers
* Collecting from the relevant [Windows Event Log](https://nxlog.co/documentation/nxlog-user-guide/windows-dns-server.html#dns_events_msvistalog) channels
* File-based [DNS debug logging](https://nxlog.co/documentation/nxlog-user-guide/windows-dns-server.html#dns_windows_setup_debug)

The deployment and resources to be used for DNS log collection will also depend on whether the logs will be collected from the DNS server (a critical asset) or from DNS clients. Each of these will be covered in further detail in this blog post.

**Collecting DNS Query Logs from Sysmon**

As of Sysmon version 10.0, there is a DNS Query logging feature to collect DNS query logs from clients. These events are generated when a process executes a DNS query, whether the result is successful or fails, cached or not.

Depending on how Sysmon is configured, you can also set additional rules in the configuration file for Sysmon in relation to[Event ID 22: DNSEvent (DNS query)](https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon#event-id-22-dnsevent-dns-query). This is advisable due to the noisy nature of this type of event. These types of additions can be:

* Exclusion rules to avoid logging reverse DNS lookups
* Exclusion rules about which domains to exclude. If excluding certain top level domains (to reduce the amount of logs collected), be more specific with domains
* Rules to exclude IPv6 lookups
* Rules to omit domains typically used in sandboxes like localhost
* Rules to omit queries involving popular third-party applications like Google, Mozilla, as well as CDNs
* Rules to omit sites that involve social media widgets like Disqus
* Rules to exclude ad serving sites and other ad-related services These are only suggestions for rules and are by all means non-exhaustive. There are Sysmon configuration samples available online for use and adaptation.

Since DNS queries generate a large amount of logs, you may opt to forward Sysmon DNS events in their own output stream to a central log server instead of merging them with other DNS client event sources.

Sysmon DNS Query Event Sample in JSON Format

{  
"EventTime":"2020-02-04T14:59:39.343541+00:00",  
"Hostname":"EC2AMAZ-EPO7HKA",  
"Keywords":"9223372036854775808",  
"EventType":"INFO",  
"SeverityValue":2,  
"Severity":"INFO",  
"EventID":22,  
"SourceName":"Microsoft-Windows-Sysmon",  
"ProviderGuid":"{5770385F-C22A-43E0-BF4C-06F5698FFBD9}",  
"Version":5,  
"TaskValue":22,  
"OpcodeValue":0,  
"RecordNumber":9532,  
"ExecutionProcessID":1996,  
"ExecutionThreadID":2616,  
"Channel":"Microsoft-Windows-Sysmon/Operational",  
"Domain":"NT AUTHORITY",  
"AccountName":"SYSTEM",  
"UserID":"S-1-5-18",  
"AccountType":"User",  
"Message":"Dns query:\r\nRuleName: \r\nUtcTime: 2020-02-04 14:59:38.349\r\nProcessGuid: {b3c285a4-3cda-5dc0-0000-001077270b00}\r\nProcessId: 1904\r\nQueryName: EC2AMAZ-EPO7HKA\r\nQueryStatus: 0\r\nQueryResults: 172.31.46.38;\r\nImage: C:\\Program Files\\nxlog\\nxlog.exe",  
"Category":"Dns query (rule: DnsQuery)",  
"Opcode":"Info",  
"UtcTime":"2020-02-04 14:59:38.349",  
"ProcessGuid":"{b3c285a4-3cda-5dc0-0000-001077270b00}",  
"ProcessId":"1904","QueryName":"EC2AMAZ-EPO7HKA","QueryStatus":"0",  
"QueryResults":"172.31.46.38;",  
"Image":"C:\\Program Files\\nxlog\\nxlog.exe",  
"EventReceivedTime":"2020-02-04T14:59:40.780905+00:00",  
"SourceModuleName":"in",  
"SourceModuleType":"im\_msvistalog"  
 }

**Collecting from DNS ETW Providers**

The DNS ETW providers with their corresponding GUIDs are displayed in the table below.

| Table 1. List of ETW Providers | |
| --- | --- |
| **ETW Provider Name** | **GUID** |
| DNS Server Trace Provider | 57840C25-FA99-4F0D-928D-D81D1851E3DD |
| Microsoft-Windows-DNS-Client | 1C95126E-7EEA-49A9-A3FE-A378B03DDB4D |
| Microsoft-Windows-DNS-Server-Service | 71A551F5-C893-4849-886B-B5EC8502641E |
| Microsoft-Windows-DNSServer | EB79061A-A566-4698-9119-3ED2807060E7 |

Most of the time, ETW is not considered as a log source, either because it is not widely known, or because special tools are needed to keep track of log traces (see [Solving Windows Log Collection Challenges with Event Tracing](https://nxlog.co/whitepapers/windows-event-tracing)). In addition, these tools can negatively affect DNS server performance, especially if they are set to continuously collect and write event traces to disk or convert to a format like JSON before being forwarded to a remote host.

**Enhanced Windows DNS Event Log Logging**

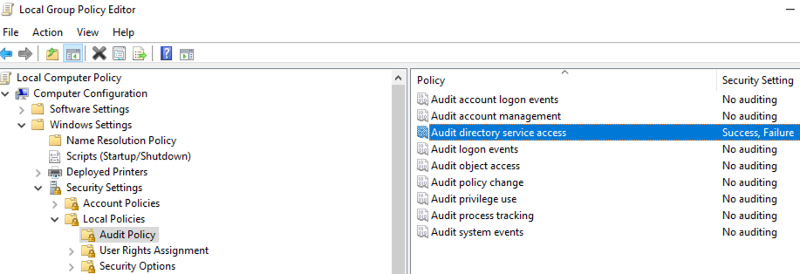
Enhanced DNS Server[audit events](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/dn800669(v=ws.11)?redirectedfrom=MSDN#audit-events) are available via both the Windows Event Log channels, such as theMicrosoft-Windows-DNSServer/Audit channel, as well as directly from the Windows Event Tracing (ETW) provider. These enable change tracking on Windows DNS Server, provided audit events are set to be logged in the Group Policy Editor. If enabled, an audit event is logged for each instance when changes are made to the DNS server such as:

**Windows DNS Audit Events**

* **Zone operations** –  zone deletions, updates, zone record creation and deletion, zone scope creation and deletion, online signing (zone signing/ unsigning/resigning), pausing/reloading/resuming zones
* **DNSSEC operations** – key rollover events, export/importing of DNSSEC metadata, addition of trust point
* **Cache operations** (cache purge events)
* **Policy operation events** – creation/deletion/updating of records such as client subnet records, server level policies or zone level policies
* **Other server operations** – restarting the server, clearing of debug logs, clearing of statistics, scavenging operations

These audit events represent important operations for any DNS server. They can provide very useful information for security and compliance reasons, as well as for incident response.

Ensure that auditing is enabled on Windows DNS Server via the Group Policy Management Editor. You can also configure auditing on the target object via the ADSIEDIT.MSC console by making the necessary changes for the auditing properties of that object.



The following is an event sample from Microsoft Windows DNS Server for the audit event 513 (Type: Zone delete, Category: Zone operations) generated by theMicrosoft-Windows-DNSServer channel.

Zone Deletion Event Sample in JSON Format

{  
 "EventTime":"2020-02-17T13:41:17.260147+00:00",  
 "Hostname":"Workstation A",  
 "Keywords":"4611686018427912192",  
 "EventType":"INFO",  
 "SeverityValue":2,  
 "Severity":"INFO",  
 "EventID":513,  
 "SourceName":"Microsoft-Windows-DNSServer",  
 "ProviderGuid":"{EB79061A-A566-4698-9119-3ED2807060E7}",  
 "Version":0,  
 "TaskValue":5,  
 "OpcodeValue":0,  
 "RecordNumber":10,  
 "ExecutionProcessID":1936,  
 "ExecutionThreadID":2820,  
 "Channel":"Microsoft-Windows-DNSServer/Audit",  
 "Domain":"Example.corp.com",  
 "AccountName":"Administrator",  
 "UserID":"S-1-5-21-211798184-3831789826-2356777772-500",  
 "AccountType":"User",  
 "Message":"The zone example.com was deleted. [virtualization instance: .].",  
 "Category":"ZONE\_OP",  
 "Opcode":"Info",  
 "Zone":"test.corp.com",  
 "VirtualizationID":".",  
 "EventReceivedTime":"2020-02-17T13:41:23.978572+00:00",  
 "SourceModuleName":"in",  
 "SourceModuleType":"im\_msvistalog"  
 }

**Windows DNS Analytical Events**

DNS[analytical events](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/dn800669(v=ws.11)?redirectedfrom=MSDN#analytic-events) differ from DNS auditing in that they are generated each time Windows DNS Server processes a request. They need to be enabled on the DNS server before logging can happen.

Types of DNS Analytical events include:

* Look up events – response success/failure, CNAME lookups, internal lookups
* Recursive query events
* Dynamic update events
* Zone XFR events

The following sample shows Event ID 280 (Type: Internal lookup additional, Category: Lookup) that is generated by ETW ProviderMicrosoft-Windows-DNSServer.

Internal Lookup Event Sample in JSON Format

{  
 "SourceName":"Microsoft-Windows-DNSServer",  
 "ProviderGuid":"{EB79061A-A566-4698-9119-3ED2807060E7}",  
 "EventId":280,  
 "Version":0,  
 "ChannelID":16,  
 "OpcodeValue":0,  
 "TaskValue":1,  
 "Keywords":"9223372105574252544",  
 "EventTime":"2020-02-15T22:30:16.466802+00:00",  
 "ExecutionProcessID":2064,  
 "ExecutionThreadID":2220,  
 "EventType":"INFO",  
 "SeverityValue":2,  
 "Severity":"INFO",  
 "Domain":"NT AUTHORITY",  
 "AccountName":"SYSTEM",  
 "UserID":"S-1-5-18",  
 "AccountType":"User",  
 "Flags":"33152",  
 "TCP":"1",  
 "InterfaceIP":"::1",  
 "Source":"::1",  
 "RD":"1",  
 "QNAME":"a.root-servers.net.",  
 "QTYPE":"28",  
 "Port":"58368",  
 "XID":"23042",  
 "BufferSize":"17",  
 "PacketData":"0x5A02818000010001000000010000060001",  
 "EventReceivedTime":"2020-02-15T22:30:17.473631+00:00",  
 "SourceModuleName":"etw",  
 "SourceModuleType":"im\_etw"  
 }

**Active Directory and Native DNS Auditing**

DNS is automatically installed with Active Directory as the Global Catalog server for the forest and domain. There are a number of features available in Windows DNS Server, such as Native DNS Auditing.

However, systems prior to 2012 R2, or 2012 R2 without hotfix 2956577 do not have native DNS auditing capabilities included. When this is enabled, DNS changes can be audited by enabling AD Directory Services auditing. For more information, see the [AD DS Auditing Step-by-Step Guide on Microsoft Docs](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc731607(v=ws.10)).

**Collecting File-based Microsoft DNS Debug Log Files**

The DNS debug file is important since it contains detailed information on DNS queries and activity that is sent and received by the DNS server.

The following debug log sample displays a simple DNS query test from Windows DNS Server:

02/12/2020 9:49:38 PM 0820 PACKET 000001D5B29C4610 UDP Snd 172.31.12.204  
 43f0 R Q [8084 A R NOERROR] PTR (1)1(1)0(1)0(3)127(7)in-addr(4)arpa(0)

Due to the amount of logs being generated from DNS debug logging, it is recommended to rotate logs and have them collected on a central server. Also, parsing the logs is suggested, in order to select which logs to enrich. Although DNS debug logging has some advantages, it does come with some additional caveats worth considering:

* Due to the way Microsoft handles log rollover of DNS debug logs, if the log file is located on any drive other than the C: drive, the Windows DNS service may not recreate the debug log file after a rollover. See[The disappearing Windows DNS debug log](https://nxlog.co/news/disappearing-windows-dns-debug-log) for an in-depth analysis of this issue.
* The log information gleaned from DNS debug logging is inherently unstructured. Parsing is required to create usable event logs. If the**Details** option has been selected, regular expressions are needed to parse the event fields. Such configurations are complex and can be associated with additional performance overhead. For busy DNS servers, this would not be a recommended option. For more information see[File-based DNS Debug Logging](https://nxlog.co/documentation/nxlog-user-guide/windows-dns-server.html#dns_windows_filebased_logging).

**Performance Considerations**

Depending on which of these logging methods you use, there are a few variables that can affect performance:

* The DNS server’s hardware specifications.
* The QPS (queries per second) rate.
* The place where log enrichment or parsing is done. It can be done either locally or on a central logging server after the logs are received.
* The type of logging taking place. It is recommended to enable DNS debug logging only temporarily as needed.

All these factors play a role in influencing log performance.

**What can NXLog do?**

NXLog simplifies DNS log collection by providing a single software solution that incorporates the various technologies required to efficiently collect DNS related logs. NXLog offers the following methods for the above discussed DNS logging technologies.

**Sysmon Log Collection**

Use the *im\_msvistalog* module and add the relevant Query in the configuration file. Find out more at[Collecting DNS logs via Sysmon](https://nxlog.co/documentation/nxlog-user-guide/windows-dns-server.html#sysmon_dns_query_logging) in the NXLog User Guide.

**ETW (Event Tracing for Windows) Collection**

There is a module,*im\_etw*, that is specifically designed to collect logs from ETW providers without much performance overhead. It acts both as a Controller and a Consumer (see[Using NXLog as a Single Agent Solution to Collect ETW Logs](https://nxlog.co/whitepapers/windows-event-tracing#using-nxlog-as-a-single-agent-solution-to-collect-etw-logs)).

**Native Windows Event Log Collection**

For DNS events that can be collected from the Windows Event Log, including Sysmon, use the *im\_msvistalog* module and specify a query for the name of the channel and channel type. You can also add additional filtering to the query. See [Windows Event Log](https://nxlog.co/documentation/nxlog-user-guide/windows-eventlog.html).

**File-based Log Collection from the Windows DNS Debug File**

There is a section in our User Guide detailing the steps involved for the setup of [DNS debug logging](https://nxlog.co/documentation/nxlog-user-guide/windows-dns-server.html#dns_windows_setup_debug) including [Parsing Non-Detailed Logs With *xm\_msdns*](https://nxlog.co/documentation/nxlog-user-guide/windows-dns-server.html#parsing-non-detailed-logs-with-xm_msdns).

**Conclusion**

With this article, you have learned about the opportunities and challenges with these modes of Windows DNS log collection: Sysmon, Event Tracing for Windows (ETW), Windows Event Log and Windows DNS debug file logging. You have also learned about possible DNS performance considerations and the solutions available for DNS log collection. With this knowledge of the various solutions available, you can avoid the pitfalls of deploying less efficient solutions, or ending up with a deployment that is either logging too many or not enough DNS events.

DNS, for many reasons, is an important asset that must not be overlooked. It is known that attackers are abusing DNS, and it is through efficient and reliable DNS log collection that you can reap the benefits of this essential component of security monitoring. Our white paper,[The Importance of DNS Logging in Enterprise Security](https://nxlog.co/whitepapers/dns-logging) expands on this theme.

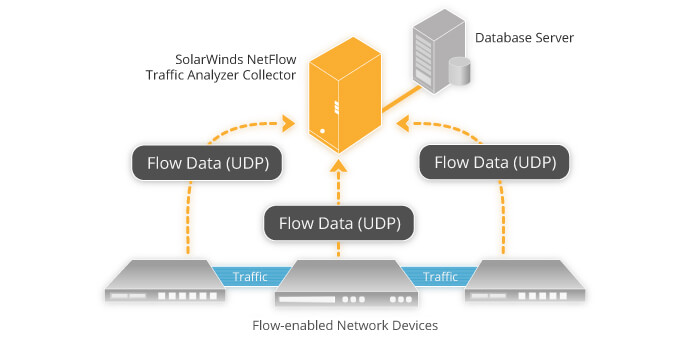
NXLog Ltd. develops multi-platform log collection tools that support many different log sources, formats, transports, and integrations. The tools help administrators collect, parse, and forward logs so they can more easily respond to security issues, investigate operational problems, and analyze event data. NXLog distributes the free and open source[NXLog Community Edition](https://nxlog.co/products/nxlog-community-edition) and offers additional features and support with the[NXLog Enterprise Edition](https://nxlog.co/products/nxlog-enterprise-edition).

This document is provided for informational purposes only and is subject to change without notice. Trademarks are the properties of their respective owners.

ref: <https://www.solarwinds.com/netflow-traffic-analyzer/use-cases/what-is-netflow>

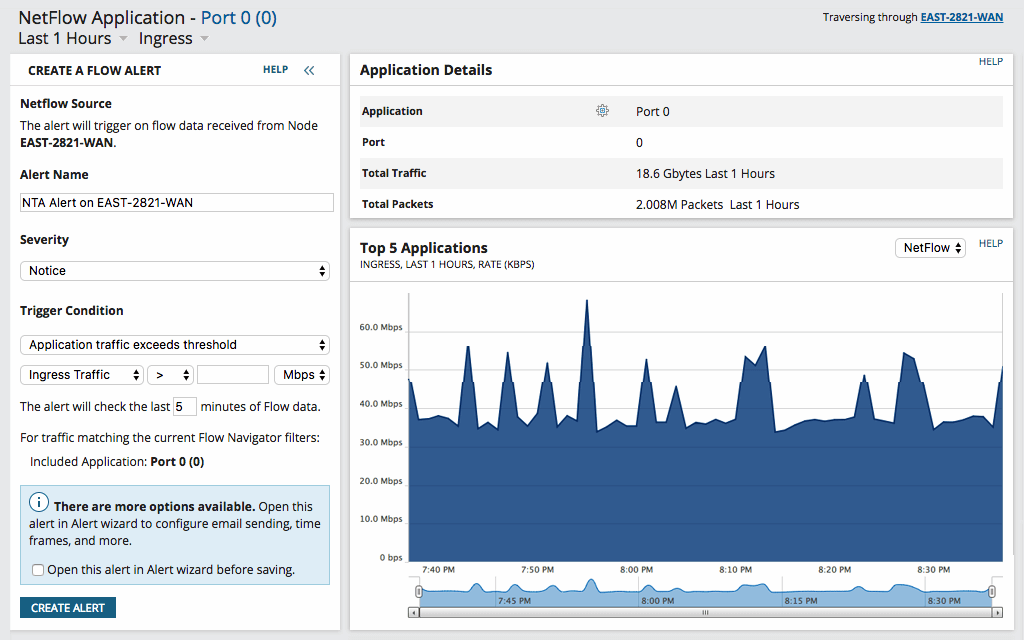
**What Is NetFlow? Get valuable insights into network and bandwidth usage**

  Fully functional for 30 days

[](https://www.solarwinds.com/-/media/solarwinds/swdcv2/licensed-products/netflow-traffic-analyzer/images/product-screenshots/graphic-netflow.ashx?rev=8f3b12e7638242168f86296b0c120b74)

[DOWNLOAD FREE TRIAL](https://www.solarwinds.com/netflow-traffic-analyzer/registration)

* [What is NetFlow?](https://www.solarwinds.com/netflow-traffic-analyzer/use-cases/what-is-netflow#anchor1)
* [Monitor NetFlow](https://www.solarwinds.com/netflow-traffic-analyzer/use-cases/what-is-netflow#anchor2)
* [Collect Data](https://www.solarwinds.com/netflow-traffic-analyzer/use-cases/what-is-netflow#anchor3)
* [Use Deep Understandings](https://www.solarwinds.com/netflow-traffic-analyzer/use-cases/what-is-netflow#anchor4)
* [Monitor Flow Alternatives](https://www.solarwinds.com/netflow-traffic-analyzer/use-cases/what-is-netflow#anchor5)

[[](https://www.solarwinds.com/-/media/solarwinds/swdcv2/licensed-products/netflow-traffic-analyzer/images/product-screenshots/nta-create-flow-alert.ashx?rev=e9775522151848a59695e07e9efe8444)](https://www.solarwinds.com/-/media/solarwinds/swdcv2/licensed-products/netflow-traffic-analyzer/images/product-screenshots/nta-create-flow-alert.ashx?rev=e9775522151848a59695e07e9efe8444)

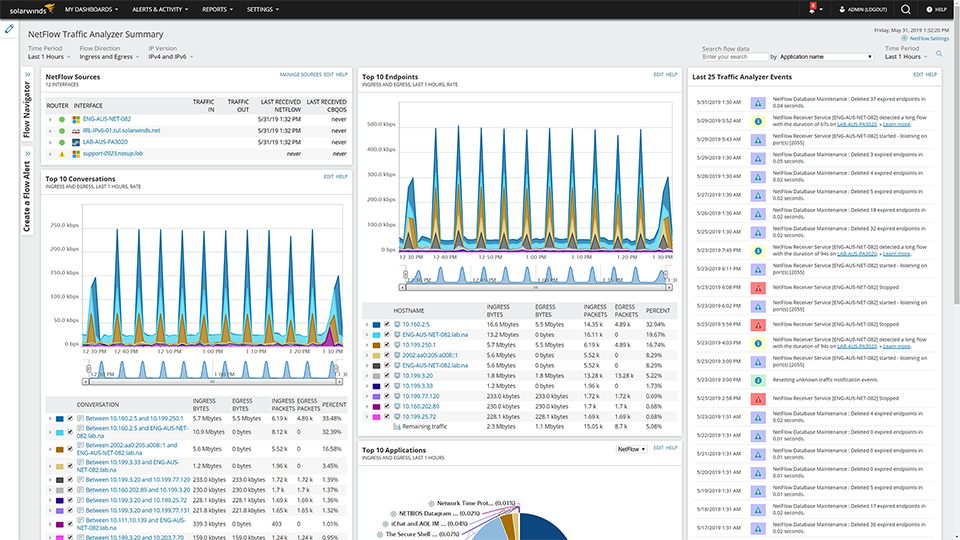
**What is NetFlow?**

NetFlow is a network protocol developed by Cisco for  collecting IP traffic information and monitoring network flow. By  analyzing NetFlow data, you can get a picture of network traffic flow  and volume.

NetFlow is a one-way technology, so when the server responds to the  initial client request, the process works in reverse and creates a new  flow record. Using a NetFlow monitoring solution can allow you to  monitor and analyze these flow records more efficiently and effectively  for traffic within the network.

SolarWinds® NetFlow Traffic Analyzer (NTA) is a powerful  and affordable NetFlow management solution with comprehensive monitoring tools designed to translate granular detail into easy-to-understand  graphs and reports—helping you more clearly identify the largest  resource drains your bandwidth.

[DOWNLOAD FREE TRIAL](https://www.solarwinds.com/netflow-traffic-analyzer/registration)  Fully functional for 30 days

[Learn More](https://www.solarwinds.com/netflow-traffic-analyzer)              

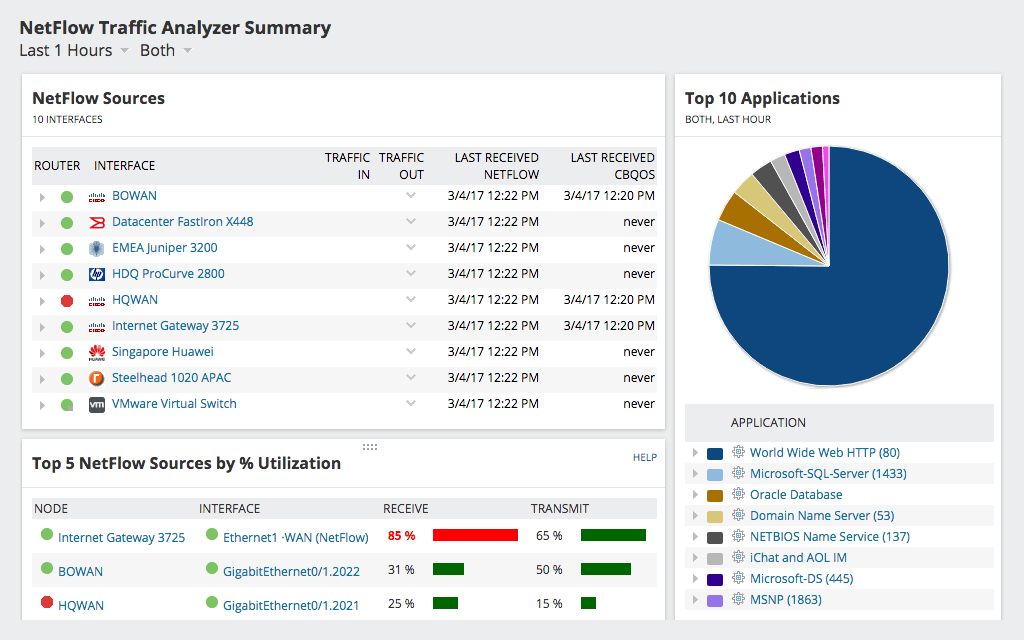
**Monitor NetFlow data more efficiently**

NetFlow data can provide valuable data about network traffic  and utilization. For effective NetFlow monitoring, a device operating as a flow exporter collates data packets into flows and sends flow records to one or more NetFlow collection servers. Then, the collectors store  and prepare the data records for analysis, which can reveal the source  and destination of a given flow record, congestion sources, and more.

SolarWinds NTA is built to combine these necessary components of a  comprehensive NetFlow monitoring system into a single, easy-to-use tool.

[DOWNLOAD FREE TRIAL](https://www.solarwinds.com/netflow-traffic-analyzer/registration)  Fully functional for 30 days

[Learn More](https://www.solarwinds.com/netflow-traffic-analyzer)

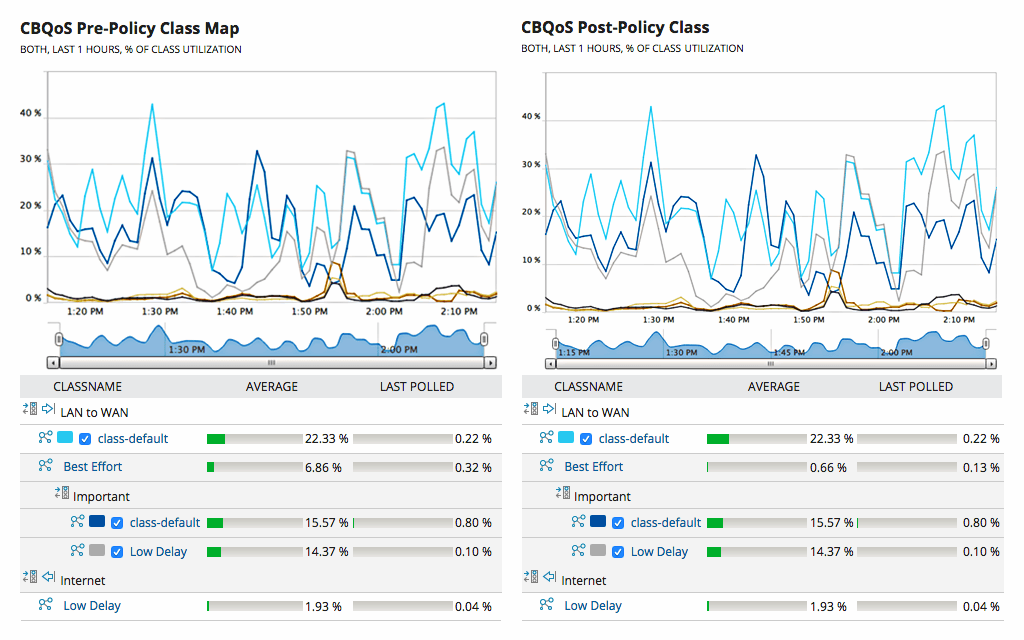
[[](https://www.solarwinds.com/-/media/solarwinds/swdcv2/licensed-products/netflow-traffic-analyzer/images/product-screenshots/nta-dashboard-summary.ashx?rev=55ae84fff0cd4e5390a95e6fea24656e)](https://www.solarwinds.com/-/media/solarwinds/swdcv2/licensed-products/netflow-traffic-analyzer/images/product-screenshots/nta-dashboard-summary.ashx?rev=55ae84fff0cd4e5390a95e6fea24656e)

**Collect data from NetFlow v5, v9, and IPFIX**

There are several data sources NTA can collect network traffic metrics from, including Cisco NetFlow v5 and v9—two of the most  commonly used network protocol systems—and NetFlow v10, more commonly  known as IPFIX. The fields that can be matched and exported are preset  in the NetFlow v5 protocol, and the template-based v9 offers more  flexibility in terms of format. IPFIX provides a standard for how IP  network flow data is formatted and transferred when exported to a  collector device. With NTA, you can monitor this kind of data—and  more—with ease.

[DOWNLOAD FREE TRIAL](https://www.solarwinds.com/netflow-traffic-analyzer/registration)  Fully functional for 30 days

[Learn More](https://www.solarwinds.com/netflow-traffic-analyzer)

[[](https://www.solarwinds.com/-/media/solarwinds/swdcv2/licensed-products/netflow-traffic-analyzer/images/product-screenshots/nta-pre-and-post.ashx?rev=edfd470b9e23477582b6a257e9fd92f5)](https://www.solarwinds.com/-/media/solarwinds/swdcv2/licensed-products/netflow-traffic-analyzer/images/product-screenshots/nta-pre-and-post.ashx?rev=edfd470b9e23477582b6a257e9fd92f5)

**Use deep understandings of network traffic to speed problem resolution**

Collecting and analyzing NetFlow data can help you understand  which users, applications, and protocols may be consuming the most  network bandwidth by tracking processes, protocols, times of day, and  traffic routing.

NTA is designed to provide a holistic view of your network traffic,  so you can more easily examine traffic patterns and monitor traffic from specific IP addresses, ports, and users to more quickly identify the  cause of bottlenecks and to support [quality of service (QoS) validation](https://www.solarwinds.com/netflow-traffic-analyzer/use-cases/network-qos-report).

[DOWNLOAD FREE TRIAL](https://www.solarwinds.com/netflow-traffic-analyzer/registration)  Fully functional for 30 days

[Learn More](https://www.solarwinds.com/netflow-traffic-analyzer)

[[](https://www.solarwinds.com/-/media/solarwinds/swdcv2/licensed-products/perfstack/images/product-screenshots/drag-n-drop.ashx?rev=457dd171d77146f4a85bcffa9c44d6cf)](https://www.solarwinds.com/-/media/solarwinds/swdcv2/licensed-products/perfstack/images/product-screenshots/drag-n-drop.ashx?rev=457dd171d77146f4a85bcffa9c44d6cf)

**Monitor flow alternatives**

In addition to automatically collecting NetFlow data from  Cisco vendors, NTA can also monitor alternative flow technologies,  including:

* Juniper (Jflow)
* 3Com/HP, Dell, and Netgear (s-flow)
* Huawei (NetStream)
* Alcatel-Lucent (Cflow)
* Ericsson (Rflow)

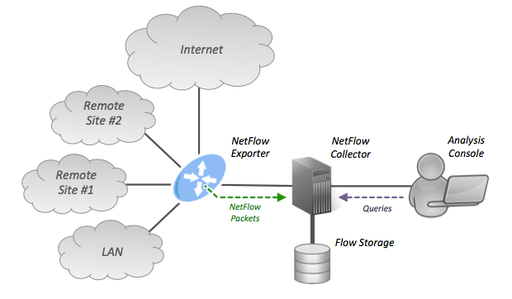
Regardless of the source of the flow data, NTA helps highlight the  applications, IP addresses, processes, protocols, and end users  consuming the most bandwidth, giving administrators valuable insights  into the behavior and performance of their networks.

ref: <https://en.wikipedia.org/wiki/NetFlow>

**NetFlow**

From Wikipedia, the free encyclopedia

[Jump to navigation](https://en.wikipedia.org/wiki/NetFlow#mw-head) [Jump to search](https://en.wikipedia.org/wiki/NetFlow#searchInput)

[](https://en.wikipedia.org/wiki/File:NetFlow_Architecture_2012.png)

NetFlow architecture

**NetFlow** is a feature that was introduced on [Cisco](https://en.wikipedia.org/wiki/Cisco) routers around 1996 that provides the ability to collect IP network  traffic as it enters or exits an interface. By analyzing the data  provided by NetFlow, a network administrator can determine things such  as the source and destination of traffic, class of service, and the  causes of congestion. A typical flow monitoring setup (using NetFlow)  consists of three main components:[[1]](https://en.wikipedia.org/wiki/NetFlow#cite_note-Flow_Monitoring_Tutorial-1)

* **Flow exporter**: aggregates packets into flows and exports flow records towards one or more flow collectors.
* **Flow collector**: responsible for reception, storage and pre-processing of flow data received from a flow exporter.
* **Analysis application**: analyzes received flow data in the context of intrusion detection or traffic profiling, for example.

**Contents**

* [1 Protocol description](https://en.wikipedia.org/wiki/NetFlow#Protocol_description)
  + [1.1 Network flows](https://en.wikipedia.org/wiki/NetFlow#Network_flows)
  + [1.2 Export of records](https://en.wikipedia.org/wiki/NetFlow#Export_of_records)
    - [1.2.1 Packet transport protocol](https://en.wikipedia.org/wiki/NetFlow#Packet_transport_protocol)
    - [1.2.2 Packet headers](https://en.wikipedia.org/wiki/NetFlow#Packet_headers)
  + [1.3 Records](https://en.wikipedia.org/wiki/NetFlow#Records)
  + [1.4 Interfaces](https://en.wikipedia.org/wiki/NetFlow#Interfaces)
  + [1.5 Sampled NetFlow](https://en.wikipedia.org/wiki/NetFlow#Sampled_NetFlow)
* [2 Versions](https://en.wikipedia.org/wiki/NetFlow#Versions)
  + [2.1 NetFlow and IPFIX](https://en.wikipedia.org/wiki/NetFlow#NetFlow_and_IPFIX)
  + [2.2 Equivalents](https://en.wikipedia.org/wiki/NetFlow#Equivalents)
  + [2.3 Support](https://en.wikipedia.org/wiki/NetFlow#Support)
* [3 Variants](https://en.wikipedia.org/wiki/NetFlow#Variants)
  + [3.1 Cisco's NetFlow Security Event Logging](https://en.wikipedia.org/wiki/NetFlow#Cisco's_NetFlow_Security_Event_Logging)
  + [3.2 Monitoring based on standalone probes](https://en.wikipedia.org/wiki/NetFlow#Monitoring_based_on_standalone_probes)
* [4 History](https://en.wikipedia.org/wiki/NetFlow#History)
* [5 See also](https://en.wikipedia.org/wiki/NetFlow#See_also)
* [6 References](https://en.wikipedia.org/wiki/NetFlow#References)
* [7 External links](https://en.wikipedia.org/wiki/NetFlow#External_links)

**Protocol description**

[Routers](https://en.wikipedia.org/wiki/Router_(computing)) and switches that support NetFlow can collect [IP](https://en.wikipedia.org/wiki/Internet_Protocol) traffic statistics on all interfaces where NetFlow is enabled, and  later export those statistics as NetFlow records toward at least one  NetFlow collector—typically a server that does the actual traffic  analysis.

**Network flows**

Cisco standard NetFlow version 5 defines a *flow* as a unidirectional sequence of packets that all share seven values which define a unique key for the flow:[[2]](https://en.wikipedia.org/wiki/NetFlow#cite_note-2)

1. Ingress interface ([SNMP](https://en.wikipedia.org/wiki/Simple_Network_Management_Protocol) ifIndex)
2. Source [IP address](https://en.wikipedia.org/wiki/IP_address)
3. Destination [IP address](https://en.wikipedia.org/wiki/IP_address)
4. [IP protocol](https://en.wikipedia.org/wiki/IP_protocol)
5. Source port for [UDP](https://en.wikipedia.org/wiki/User_Datagram_Protocol) or [TCP](https://en.wikipedia.org/wiki/Transmission_Control_Protocol), 0 for other protocols
6. Destination port for [UDP](https://en.wikipedia.org/wiki/User_Datagram_Protocol) or [TCP](https://en.wikipedia.org/wiki/Transmission_Control_Protocol), type and code for [ICMP](https://en.wikipedia.org/wiki/Internet_Control_Message_Protocol), or 0 for other protocols
7. IP [Type of Service](https://en.wikipedia.org/wiki/Type_of_Service)

Note that the Egress interface, IP Nexthop or BGP Nexthops are not  part of the key, and may not be accurate if the route changes before the expiration of the flow, or if load-balancing is done per-packet.

This definition of flows is also used for IPv6, and a similar definition is used for [MPLS](https://en.wikipedia.org/wiki/MPLS) and [Ethernet](https://en.wikipedia.org/wiki/Layer_2) flows.

Advanced NetFlow or IPFIX implementations like Cisco Flexible NetFlow allow user-defined flow keys.

A typical output of a NetFlow command line tool (nfdump in this case) when printing the stored flows may look as follows:

Date flow start Duration Proto Src IP Addr:Port Dst IP Addr:Port Packets Bytes Flows  
 2010-09-01 00:00:00.459 0.000 UDP 127.0.0.1:24920 -> 192.168.0.1:22126 1 46 1  
 2010-09-01 00:00:00.363 0.000 UDP 192.168.0.1:22126 -> 127.0.0.1:24920 1 80 1

**Export of records**

The router will output a flow record when it determines that the flow is  finished.  It does this by flow aging: when the router sees new traffic  for an existing flow it resets the aging counter. Also, [TCP session](https://en.wikipedia.org/w/index.php?title=TCP_session&action=edit&redlink=1) termination in a TCP flow causes the router to expire the flow. Routers can also be configured to output a flow record at a fixed interval even if the flow is still ongoing.

Packet transport protocol

NetFlow records are traditionally exported using User Datagram Protocol ([UDP](https://en.wikipedia.org/wiki/User_Datagram_Protocol)) and collected using a NetFlow collector.  The IP address of the NetFlow collector and the destination UDP port  must be configured on the sending router. A common value is UDP port  2055, but other values like 9555 or 9995, 9025, 9026 etc. can also be  used.

For efficiency reasons, the router traditionally does not keep  track of flow records already exported, so if a NetFlow packet is  dropped due to [network congestion](https://en.wikipedia.org/wiki/Network_congestion) or packet corruption, all contained records are lost forever. The UDP  protocol does not inform the router of the loss so it can send the  packets again. This can be a real problem, especially with NetFlow v8 or v9 that can  aggregate a lot of packets or flows into a single record. A single UDP  packet loss can cause a huge impact on the statistics of some flows.

That is why some modern implementations of NetFlow use the Stream Control Transmission Protocol ([SCTP](https://en.wikipedia.org/wiki/Stream_Control_Transmission_Protocol)) to export packets so as to provide some protection against packet loss, and make sure that NetFlow v9 templates are received before any related record is exported. Note that TCP would not be suitable for NetFlow  because a strict ordering of packets would cause excessive buffering and delays.

The problem with SCTP is that it requires interaction between  each NetFlow collector and each routers exporting NetFlow. There may be  performance limitations if a router has to deal with many NetFlow  collectors, and a NetFlow collector has to deal with many routers,  especially when some of them are unavailable due to failure or  maintenance.

SCTP may not be efficient if NetFlow must be exported toward  several independent collectors, some of which may be test servers that  can go down at any moment. UDP allows simple replication of NetFlow packets using Network taps or  L2 or L3 Mirroring. Simple stateless equipment can also filter or change the destination address of NetFlow UDP packets if necessary. Since  NetFlow export almost only use network backbone links, packet loss will  often be negligible. If it happens, it will mostly be on the link  between the network and the NetFlow collectors.

Packet headers

All NetFlow packets begin with version-dependent header, that contains at least these fields:

* Version number (v1, v5, v7, v8, v9)
* Sequence number to detect loss and duplication
* Timestamps at the moment of export, as system uptime or absolute time.
* Number of records (v5 or v8) or list of templates and records (v9)

**Records**

A NetFlow record can contain a wide variety of information about the traffic in a given flow.

NetFlow version 5 (one of the most commonly used versions, followed by version 9) contains the following:

* Input interface index used by [SNMP](https://en.wikipedia.org/wiki/Simple_Network_Management_Protocol) (ifIndex in IF-MIB).
* Output interface index or zero if the packet is dropped.
* Timestamps for the flow start and finish time, in milliseconds since the last boot.
* Number of bytes and packets observed in the flow
* [Layer 3](https://en.wikipedia.org/wiki/Network_Layer) headers:
  + Source & destination IP addresses
  + [ICMP](https://en.wikipedia.org/wiki/Internet_Control_Message_Protocol) Type and Code.
  + IP protocol
  + [Type of Service](https://en.wikipedia.org/wiki/Type_of_Service) (ToS) value
* Source and destination port numbers for TCP, UDP, SCTP
* For TCP flows, the union of all TCP flags observed over the life of the flow.
* Layer 3 [Routing](https://en.wikipedia.org/wiki/Routing) information:
  + IP address of the immediate next-hop (not the BGP nexthop) along the route to the destination
  + Source & destination IP masks (prefix lengths in the [CIDR](https://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing) notation)

For [ICMP](https://en.wikipedia.org/wiki/Internet_Control_Message_Protocol) flows, the Source Port is zero, and the Destination Port number field  codes ICMP message Type and Code (port = ICMP-Type \* 256 + ICMP-Code).

The source and destination [Autonomous System](https://en.wikipedia.org/wiki/Autonomous_system_(Internet)) (AS) number fields can report the destination AS (last AS of AS-Path)  or the immediate neighbor AS (first AS of AS-Path) depending on the  router configuration. But the AS number will be zero if the feature is  not supported, the route is unknown or not announced by BGP, or the AS  is the local AS. There is no explicit way to distinguish between these  cases.

NetFlow version 9 can include all of these fields and can optionally include additional information such as [Multiprotocol Label Switching](https://en.wikipedia.org/wiki/Multiprotocol_Label_Switching) (MPLS) labels and [IPv6](https://en.wikipedia.org/wiki/IPv6) addresses and ports,

By analyzing flow data, a picture of traffic flow and traffic  volume in a network can be built. The NetFlow record format has evolved  over time, hence the inclusion of version numbers. Cisco maintains  details of the different version numbers and the layout of the packets  for each version.

**Interfaces**

NetFlow is usually enabled on a per-interface basis to limit load on the router components involved in NetFlow, or to limit the amount of NetFlow  records exported.

NetFlow usually captures all packets received by an ingress IP  interface, but some NetFlow implementations use IP filters to decide if a packet can be observed by NetFlow.

Some NetFlow implementations also allow the observation of  packets on the egress IP interface, but this must be used with care: all flows from any ingress interface with NetFlow enabled to any interface  with NetFlow enabled could be counted twice.

**Sampled NetFlow**

Standard NetFlow was designed to process all IP packets on an interface.  But in some environments, e.g. on Internet backbones, that was too costly, due to the extra processing required for each packet, and large number of  simultaneous flows.

So Cisco introduced sampled NetFlow on Cisco [12000](https://en.wikipedia.org/wiki/Cisco_12000), and that is now used in all high-end routers that implement NetFlow.

Only one packet out of *n* is processed, where *n*, the sampling rate, is determined by the router configuration.

The exact selection process depends on the implementation:

* One packet every *n* packet, in Deterministic NetFlow, as used on Cisco's [12000](https://en.wikipedia.org/wiki/Cisco_12000).
* One packet randomly selected in an interval of *n* packet, in Random Sampled NetFlow, used on modern Cisco routers.

Some implementations have more complex methods to sample packets, like per-flow sampling on Cisco Martinez Catalysts.

The sampling rate is often the same for all interfaces, but can  be adjusted per interface for some routers. When Sampled NetFlow is used, the NetFlow records must be adjusted for  the effect of sampling - traffic volumes, in particular, are now an  estimate rather than the actual measured flow volume.

The sampling rate is indicated in a header field of NetFlow  version 5 (same sampling rate for all interfaces) or in option records  of NetFlow version 9 (sampling rate per interface)

**Versions**

| **Version** | **Comment** |
| --- | --- |
| **v1** | First implementation, now obsolete, and restricted to [IPv4](https://en.wikipedia.org/wiki/IPv4) (without [IP mask](https://en.wikipedia.org/wiki/CIDR_notation) and [AS Numbers](https://en.wikipedia.org/wiki/Autonomous_system_(Internet))). |
| **v2** | Cisco internal version, never released. |
| **v3** | Cisco internal version, never released. |
| **v4** | Cisco internal version, never released. |
| **v5** | Most common version, available (as of 2009) on many routers from different brands, but restricted to [IPv4](https://en.wikipedia.org/wiki/IPv4) flows. |
| **v6** | No longer supported by Cisco. Encapsulation information (?). |
| **v7** | Like version 5 with a source router field. Used (only?) on Cisco Catalyst switches. |
| **v8** | Several aggregation form, but only for information that is already present in version 5 records |
| **v9** | Template Based, available (as of 2009) on some recent routers. Mostly used to report flows like [IPv6](https://en.wikipedia.org/wiki/IPv6), [MPLS](https://en.wikipedia.org/wiki/Multiprotocol_Label_Switching), or even plain [IPv4](https://en.wikipedia.org/wiki/IPv4) with BGP nexthop. |
| **v10** | Used for identifying [IPFIX](https://en.wikipedia.org/wiki/IP_Flow_Information_Export). Although IPFIX is heavily based on NetFlow, v10 does not have anything to do with NetFlow. |

**NetFlow and IPFIX**

NetFlow was initially implemented by Cisco, and described in an "informational" document that was not on the standards track: RFC 3954 – Cisco Systems  NetFlow Services Export Version 9.  The NetFlow protocol itself has been superseded by Internet Protocol Flow Information eXport ([IPFIX](https://en.wikipedia.org/wiki/IPFIX)). Based on the NetFlow Version 9 implementation, IPFIX is on the IETF  standards track with RFC 5101 (obsoleted by RFC 7011), RFC 5102  (obsoleted by RFC 7012), etc. which were published in 2008.

**Equivalents**

Many vendors other than [Cisco](https://en.wikipedia.org/wiki/Cisco_Systems) provide similar network flow monitoring technology.  NetFlow may be a  prevalent name in the area of flow monitoring, because of  [Cisco](https://en.wikipedia.org/wiki/Cisco_Systems) dominant market share in the networking industry.  NetFlow is thought  to be a Cisco trademark (even though as of March 2012 it is not listed  in Cisco Trademarks[[3]](https://en.wikipedia.org/wiki/NetFlow#cite_note-3)):

* [Argus - Audit Record Generation and Utilization System](https://en.wikipedia.org/wiki/Argus_-_Audit_Record_Generation_and_Utilization_System)
* Jflow or cflowd for [Juniper Networks](https://en.wikipedia.org/wiki/Juniper_Networks)
* NetStream for [3Com/HP](https://en.wikipedia.org/wiki/3com)
* NetStream for [Huawei Technologies](https://en.wikipedia.org/wiki/Huawei_Technologies)
* Cflowd for [Nokia](https://en.wikipedia.org/wiki/Nokia)
* Rflow for [Ericsson](https://en.wikipedia.org/wiki/Ericsson)
* AppFlow [Citrix](https://en.wikipedia.org/wiki/Citrix)
* [sFlow](https://en.wikipedia.org/wiki/SFlow) vendors include: [Alaxala](https://en.wikipedia.org/wiki/Alaxala), [Alcatel Lucent](https://en.wikipedia.org/wiki/Alcatel_Lucent), [Allied Telesis](https://en.wikipedia.org/wiki/Allied_Telesis), [Arista Networks](https://en.wikipedia.org/wiki/Arista_Networks), [Brocade](https://en.wikipedia.org/wiki/Brocade_Communications_Systems), [Cisco](https://en.wikipedia.org/wiki/Cisco_Systems), [Dell](https://en.wikipedia.org/wiki/Dell), [D-Link](https://en.wikipedia.org/wiki/D-Link), [Enterasys](https://en.wikipedia.org/wiki/Enterasys), [Extreme](https://en.wikipedia.org/wiki/Extreme_Networks), [F5 BIG-IP](https://en.wikipedia.org/wiki/F5_Networks), [Fortinet](https://en.wikipedia.org/wiki/Fortinet), [Hewlett-Packard](https://en.wikipedia.org/wiki/Hewlett-Packard), [Hitachi](https://en.wikipedia.org/wiki/Hitachi), [Huawei](https://en.wikipedia.org/wiki/Huawei), [IBM](https://en.wikipedia.org/wiki/IBM), [Juniper](https://en.wikipedia.org/wiki/Juniper_Networks), [LG-Ericsson](https://en.wikipedia.org/wiki/LG-Ericsson), [Mellanox](https://en.wikipedia.org/wiki/Mellanox), [MRV](https://en.wikipedia.org/wiki/MRV_Communications), [NEC](https://en.wikipedia.org/wiki/NEC), [Netgear](https://en.wikipedia.org/wiki/Netgear), [Proxim Wireless](https://en.wikipedia.org/wiki/Proxim_Wireless), [Quanta Computer](https://en.wikipedia.org/wiki/Quanta_Computer), [Vyatta](https://en.wikipedia.org/wiki/Vyatta), [Telesoft](https://en.wikipedia.org/wiki/Telesoft_Technologies), [ZTE](https://en.wikipedia.org/wiki/ZTE) and [ZyXEL](https://en.wikipedia.org/wiki/ZyXEL)[[4]](https://en.wikipedia.org/wiki/NetFlow#cite_note-sFlow_Vendors-4)

Also flow-tools collection of software[[5]](https://en.wikipedia.org/wiki/NetFlow#cite_note-5) allows to process and manage NetFlow exports from Cisco and Juniper routers.[[6]](https://en.wikipedia.org/wiki/NetFlow#cite_note-6)

**Support**

| **Vendor and type** | **Models** | **NetFlow Version** | **Implementation** | **Comments** |
| --- | --- | --- | --- | --- |
| **Cisco IOS-XR routers** | [CRS](https://en.wikipedia.org/wiki/Carrier_Routing_System), [ASR9000](https://en.wikipedia.org/wiki/ASR9000) old [12000](https://en.wikipedia.org/wiki/Cisco_12000) | v5, v8, v9 | Software running on line card CPU | Comprehensive support for IPv6 and MPLS |
| **Cisco IOS routers** | 10000, 7200, old 7500 | v5, v8, v9 | Software running on Route Processor | support for IPv6 or MPLS require recent model and IOS |
| **Cisco**[**Catalyst**](https://en.wikipedia.org/wiki/Cisco_Catalyst)**switches** | 7600, 6500, 4500 | v5, v8, v9 | Dedicated hardware TCAM, also used for ACLs. | Support for IPv6 on high-end models RSP720 and Sup720, but at most 128K or 256K flows per PCF card.[[7]](https://en.wikipedia.org/wiki/NetFlow#cite_note-7) |
| **Cisco**[**Nexus**](https://en.wikipedia.org/wiki/Cisco_Nexus)**switches** | 5600, 7000, 7700 | v5, v9 | Dedicated hardware TCAM, also used for ACLs. Up to 512K flows. Support IPv4/IPv6/L2. | MPLS not supported |
| **Juniper legacy routers** | [M-series](https://en.wikipedia.org/wiki/Juniper_M-Series), [T-series](https://en.wikipedia.org/wiki/Juniper_MX-Series), [MX-series](https://en.wikipedia.org/wiki/Juniper_MX-Series) with DPC | v5, v8 | Software running on Routing Engine, called *software jflow* | IPv6 and MPLS not supported |
| **Juniper legacy routers** | [M-series](https://en.wikipedia.org/wiki/Juniper_M-Series), [T-series](https://en.wikipedia.org/wiki/Juniper_MX-Series), [MX-series](https://en.wikipedia.org/wiki/Juniper_MX-Series) with DPC | v5, v8, v9 | Software running on service PIC, called *hardware jflow* or *sampled* | IPv6 or MPLS supported on MS-DPC, MultiService-PIC, AS-PIC2 |
| [**Juniper**](https://en.wikipedia.org/wiki/Juniper_Networks)**routers** | [MX-series](https://en.wikipedia.org/wiki/Juniper_MX-Series) with MPC-3D, FPC5 for T4000 | v5, [IPFIX](https://en.wikipedia.org/wiki/IP_Flow_Information_Export) | Hardware (trio chipset), called *inline jflow* | IPv6 requires JUNOS 11.4R2 (back port target), MPLS support unknown, MPC3E excluded until 12.3, incorrect start time field causing incorrect data throughput result [[8]](https://en.wikipedia.org/wiki/NetFlow#cite_note-8) |
| [**Nokia**](https://en.wikipedia.org/wiki/Nokia)**routers** | 7750SR | v5, v8, v9, v10 [IPFIX](https://en.wikipedia.org/wiki/IP_Flow_Information_Export) | Software running on Central Processor Module | IPv6 or MPLS using IOM3 line cards or better |
| [**Huawei**](https://en.wikipedia.org/wiki/Huawei)**routers** | NE5000E NE40E/X NE80E | v5, v9 | Software running on service cards | Support for IPv6 or MPLS is unknown |
| [**Enterasys**](https://en.wikipedia.org/wiki/Enterasys_Networks)**Switches** | S-Serie[[9]](https://en.wikipedia.org/wiki/NetFlow#cite_note-9)  and N-Serie[[10]](https://en.wikipedia.org/wiki/NetFlow#cite_note-10) | v5, v9 | Dedicated hardware | IPv6 support is unknown |
| [**Flowmon**](https://en.wikipedia.org/wiki/Flowmon)**Probes** | [Flowmon](https://en.wikipedia.org/wiki/Flowmon) Probe 1000, 2000, 4000, 6000, 10000, 20000, 40000, 80000, 100000 | v5, v9, [IPFIX](https://en.wikipedia.org/wiki/IP_Flow_Information_Export) | Software or hardware-accelerated | Comprehensive support for IPv6 and MPLS, wire-speed |
| [**Nortel**](https://en.wikipedia.org/wiki/Nortel)**Switches** | Ethernet Routing Switch 5500 Series (ERS5510, 5520 and 5530) and 8600 (Chassis-based) | v5, v9, IPFIX | Software running on line card CPU | Comprehensive support for IPv6 |
| **PC and Servers** | [Linux](https://en.wikipedia.org/wiki/Linux) [FreeBSD](https://en.wikipedia.org/wiki/FreeBSD) [NetBSD](https://en.wikipedia.org/wiki/NetBSD) [OpenBSD](https://en.wikipedia.org/wiki/OpenBSD) | v5, v9, IPFIX | Software like fprobe,[[11]](https://en.wikipedia.org/wiki/NetFlow#cite_note-11) ipt-netflow,[[12]](https://en.wikipedia.org/wiki/NetFlow#cite_note-12) pflow,[[13]](https://en.wikipedia.org/wiki/NetFlow#cite_note-13) flowd,[[14]](https://en.wikipedia.org/wiki/NetFlow#cite_note-14) [Netgraph](https://en.wikipedia.org/wiki/Netgraph) ng\_netflow[[15]](https://en.wikipedia.org/wiki/NetFlow#cite_note-15) or softflowd | IPv6 support depend on the software used |
| **VMware servers** | [vSphere](https://en.wikipedia.org/wiki/VSphere) 5.x[[16]](https://en.wikipedia.org/wiki/NetFlow#cite_note-16) | v5, IPFIX (>5.1)[[17]](https://en.wikipedia.org/wiki/NetFlow#cite_note-17) | Software | IPv6 support is unknown |
| **Mikrotik RouterOS** | RouterOS 3.x, 4.x, 5.x, 6.x [[18]](https://en.wikipedia.org/wiki/NetFlow#cite_note-18) | v1, v5, v9, IPFIX (>6.36RC3) | Software and Routerboard hardware | IPv6 is supported using v9. Currently RouterOS does not include BGP AS numbers. |

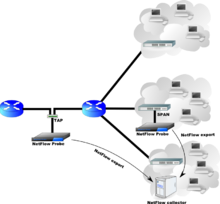
**Variants**

**Cisco's NetFlow Security Event Logging**

Introduced with the launch of the [Cisco ASA](https://en.wikipedia.org/wiki/Cisco_ASA) 5580 products, [NetFlow Security Event Logging](http://www.cisco.com/en/US/docs/security/asa/asa81/netflow/netflow.html) utilizes NetFlow v9 fields and templates in order to efficiently  deliver security telemetry in high performance environments.  NetFlow  Security Event Logging scales better than [syslog](https://en.wikipedia.org/wiki/Syslog) while offering the same level of detail and granularity in logged events.[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

**Monitoring based on standalone probes**

|  |  |
| --- | --- |
|  | This section **possibly contains**[**original research**](https://en.wikipedia.org/wiki/Wikipedia:No_original_research). Please [improve it](https://en.wikipedia.org/w/index.php?title=NetFlow&action=edit) by [verifying](https://en.wikipedia.org/wiki/Wikipedia:Verifiability) the claims made and adding [inline citations](https://en.wikipedia.org/wiki/Wikipedia:Citing_sources#Inline_citations). Statements consisting only of original research should be removed.  *(March 2009) (*[*Learn how and when to remove this template message*](https://en.wikipedia.org/wiki/Help:Maintenance_template_removal)*)* |

[](https://en.wikipedia.org/wiki/File:NewNetFlowApproach.png)

NetFlow architecture using standalone probes.

NetFlow collection using standalone NetFlow probes is an alternative  to flow collection from routers and switches. This approach can overcome some limitations of router-based NetFlow monitoring. The probes are  transparently connected to the monitored link as a passive appliance  using the TAP or SPAN port of the appliance.

Historically, NetFlow monitoring is easier to implement in a  dedicated probe than in a router. However, this approach also has some  drawbacks:

* probes must be deployed on every link that must be observed, causing additional hardware, setup and maintenance costs.
* probes will not report separate input and output interface information like a report from a router would.
* probes may have problems reporting reliably the NetFlow fields related to routing, like [AS Numbers](https://en.wikipedia.org/wiki/Autonomous_system_(Internet)) or [IP masks](https://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing), because they can hardly be expected to use exactly the same routing information as a router.

The easiest way to address the above drawbacks is to use a [packet capture appliance](https://en.wikipedia.org/wiki/Packet_capture_appliance) inline in front of the router and capture all of the NetFlow output  from the router. This method allows for storage of large amount of  NetFlow data (typically many years worth of data) and does not require  reconfiguration of the network.

NetFlow collection from dedicated probes is well suited for  observation of critical links, whereas NetFlow on routers provides a  Network-wide view of the traffic that can be used for capacity planning, accounting, performance monitoring, and security.

**History**

NetFlow was originally a Cisco packet switching technology for Cisco routers, implemented in [IOS](https://en.wikipedia.org/wiki/Cisco_IOS) 11.x around 1996. It was originally a software implementation for the Cisco 7000, 7200 and 7500,[[19]](https://en.wikipedia.org/wiki/NetFlow#cite_note-netflow_switching-19) where it was thought as an improvement over the then current Cisco Fast Switching. Netflow was invented by Darren Kerr and Barry Bruin [[20]](https://en.wikipedia.org/wiki/NetFlow#cite_note-20)from Cisco (U.S. patent # 6,243,667).

The idea was that the first packet of a flow would create a  NetFlow switching record. This record would then be used for all later  packets of the same flow, until the expiration of the flow. Only the  first packet of a flow would require an investigation of the route table to find the most specific matching route. This is an expensive  operation in software implementations, especially the old ones without [Forwarding information base](https://en.wikipedia.org/wiki/Forwarding_information_base). The NetFlow switching record was actually some kind of route cache  record, and old versions of IOS still refer to the NetFlow cache as **ip route-cache**.

This technology was advantageous for local networks. This was especially true if some of the traffic had to be filtered by an [ACL](https://en.wikipedia.org/w/index.php?title=Standard_Access_Control_List&action=edit&redlink=1) as only the first packet of a flow had to be evaluated by the ACL.[[21]](https://en.wikipedia.org/wiki/NetFlow#cite_note-kentik-21)

NetFlow switching soon turned out to be unsuitable for big  routers, especially Internet backbone routers, where the number of  simultaneous flows was much more important than those on local networks, and where some traffic causes many short-lived flows, like [Domain Name System](https://en.wikipedia.org/wiki/Domain_Name_System) requests (whose source port is random for security reasons).

As a switching technology, NetFlow was replaced around 1995 by [Cisco Express Forwarding](https://en.wikipedia.org/wiki/Cisco_Express_Forwarding). This first appeared on Cisco 12000 routers, and later replaced NetFlow  switching on advanced IOS for the Cisco 7200 and Cisco 7500.

As of 2012, technologies similar to NetFlow switching are still  in use in most firewalls and software-based IP routers. For instance the conntrack feature of the [Netfilter](https://en.wikipedia.org/wiki/Netfilter) framework used by [Linux](https://en.wikipedia.org/wiki/Linux).

Ref: <https://www.vanimpe.eu/2016/10/21/proxy-server-logs-incident-response/>

[**Proxy server logs for incident response**](https://www.vanimpe.eu/2016/10/21/proxy-server-logs-incident-response/)

 Posted on [October 21, 2016](https://www.vanimpe.eu/2016/10/21/proxy-server-logs-incident-response/)    in [internet](https://www.vanimpe.eu/category/internet/), [security](https://www.vanimpe.eu/category/security/)     

**Proxy server logs for incident response**

When you do incident response having access to detailed logs is crucial. One of those treasure troves are **proxy server logs**.

Proxy server logs contain the requests made by users and applications on your network. This does not only include the most obvious part : web site request by users but also application or service requests made to  the internet (for example application updates).

Ideally you have a transparent proxy, meaning that all outgoing  requests are redirected by a firewall to a proxy. Unfortunately not all  applications behave properly when they have to go through a proxy. As a  result, in a lot of corporate environment you’ll find the use of a proxy being forced to users or applications via a configuration setting. If  you’re using PAC files for proxy configuration then now might be a good  time to read the notification on [Proxy auto-config (PAC) files have access to full HTTPS URLs](https://www.kb.cert.org/vuls/id/877625).

It would be a shame if you have proxy server logs for incident  response only to find out that they do not contain the information that  you need during an investigation. This post contains some of the  settings you should take into consideration when configuring your proxy  server.

**Configuring proxy server logs for incident response**

**Time synchronization**

If you try to reconstruct a timeline then correct timestamps are crucial. So make sure that your proxy server is **NTP-synchronized**. Also make note of the timezone being used for logging. Ideally you use UTC.

**Log retention**

A lot of security incidents are detected long after the initial  compromise took place. If you can afford the storage you should keep  proxy logs for a relatively long time (this means years, not weeks or  months). If you don’t have enough storage you can include logs in the  backup procedure and restore them if you conduct an investigation. Make  sure that logs (and backups) are properly protected (access and  integrity). According to [Mandiant](https://www2.fireeye.com/rs/848-DID-242/images/Mtrends2016.pdf) the median number of days that attackers were present on a victim’s network is **146** days (320 days for data breaches with external notification and 56 days with internal discovery).

**Proxy log settings**

Proxy server logs should track the below information for being useful during an investigation :

* Date and time
* HTTP protocol version
* HTTP request method
* Content type
* User agent
* HTTP referer
* Length of the content response
* Authenticated username of the client
* Client IP and source port
* Target host IP and destination port
* Target hostname (DNS)
* The requested resource
* HTTP status code of reply
* Time needed to provide the reply back to the client
* Proxy action (from cache, not from cache, …)

**Alerts on proxy server entries**

Besides being useful during an incident you can also raise alerts based on the content of the proxy server logs.

**Unusual protocol version**

Most modern clients will now use HTTP/1.1. Requests with HTTP/1.0  require deeper inspection. Don’t be alarmed immediately, some older  applications might just not support HTTP/1.0. Keep a list of those  applications to exclude them from raising an alert.

**User agents**

You should not blindly trust user agent information, it’s something  that can easily be crafted. But making statistics on the user agents can prove useful. Look out for user agents that indicate the use of a  scripting language (Python for example) or user agents that don’t make  sense. You can use [User Agent String.com](http://www.useragentstring.com/pages/useragentstring.php) as a reference.

If you control your environment then you can develop a list of  “known” and “accepted” user agents. Everything that’s out of the  ordinary should then trigger an alarm.

If your proxy server logs the computer name you can add this as an  extra rule to validate the trustworthiness of the user agent field.

**HTTP request methods**

Log the HTTP request method (for example GET, POST) and graph / alert on (an increase of) unusual methods (for example CONNECT, PUT)

Focus on POSTs with content types different than text/html. Especially POSTS with application/octet-stream or any of the [MS Office document file types](https://technet.microsoft.com/en-us/library/ee309278(office.12).aspx) should raise suspicion. Repeated requests can indicate that something or someone is uploading a lot of (corporate?) documents.

GET requests contain the query string in the URL. This can easily be  logged. POST requests however have the query string in the HTTP message  body. This is not always straightforward to log. But without this  information it’s sometimes very difficult for getting to know the actual payload that was exchanged. You’ll have to look into something similar  as [mod security for logging HTTP POST requests](https://www.vanimpe.eu/2015/11/20/logging-nfsen-queries/). Also don’t forget that logging the entire query string, regardless of  GET or POST can raise privacy concerns. Consult the HR and Legal  department for advice.

**Length of the content response**

Track the length of the content response. A host that repeatedly  sends or receives the same length of content responses might indicate a  host that requires further inspection. It can mean an application update but also malware beaconing out to control servers.

Also, excessive content lengths should raise an alarm.

**Target host IP, destination port, hostname and requested resource**

Requests that go to non standard HTTP or HTTPS ports should always raise an alert.

Last but not least you should use the information provided by threat information platforms like for example [MISP](https://github.com/MISP/MISP) to track requests for hosts or resources that are known to be bad.

As bonus you can also use [passive dns information](https://www.vanimpe.eu/2016/02/27/passive-dns-for-incident-response/) in addition to inspecting the requested resources. This becomes  especially useful if your proxy servers logs both target IP and  hostname. If a domain was hosting something malicious on a specific IP  during a limited timeframe you can use both sets of data to check if you were affected.

**Collecting proxy server logs**

If you are using a BlueCoat proxy then you can use the article [BlueCoat Proxy log search and analytics with ELK](http://christophe.vandeplas.com/2014/07/bluecoat-proxy-log-search-and-analytics.html) as a guideline on how to use ELK to analyse those logs.

ref: <https://www.exabeam.com/siem-guide/siem-concepts/firewall-logs/>

**The Significance and Role of Firewall logs**

A firewall monitors traffic into and out of the environment  it was developed to protect. Some firewalls also offer visibility into  the source and type of traffic coming into this environment. A firewall  is configured using rules. To be successful, the ruleset of a firewall  must be augmented with a successful logging feature.

The logging feature records how the firewall manages traffic types.  The logs provide organizations with information about, for example,  source and destination IP addresses, protocols, and port numbers and can be used by a [SIEM](https://www.exabeam.com/siem-guide/) to help investigate an attack.

**In this page:**

* [Firewall logging](https://www.exabeam.com/siem-guide/siem-concepts/firewall-logs/#logging)
* [Linux firewall logs](https://www.exabeam.com/siem-guide/siem-concepts/firewall-logs/#linux)
* [Windows firewall logs](https://www.exabeam.com/siem-guide/siem-concepts/firewall-logs/#windows)
* [How to analyze firewall logs](https://www.exabeam.com/siem-guide/siem-concepts/firewall-logs/#analyze)
* [Log analysis and alerting with Exabeam](https://www.exabeam.com/siem-guide/siem-concepts/firewall-logs/#exabeam)

**Firewall Logging**

A firewall, at its most basic form, is created to stop connections  from suspicious networks. It inspects the source address, destination  address, and the destination port of all connections, and decides if a  network can be trusted.

For simplicity, we can aggregate information on the source address,  source port, and destination address and port. We can view this  information as an identifying quality of any attempt to connect, as  tracked by the firewall.

This quality is similar to a set of rules that determine which  connections are permitted and which must be denied. If this identifying  quality holds information that is the same as a permitted connection,  then the source address can create a connection with the destination  address on the permitted port. Thus, the traffic is allowed into the  network.

The success of any firewall, therefore, typically relies on the rules used to configure it.

The firewall monitors traffic into and out of the environment it was  created to safeguard and provides visibility into the type and source of traffic entering this environment. It typically serves two purposes:

* Protecting the environment from threats from internal and external (Internet) sources
* Acting as an investigative resource for security professionals who need to track how a breach penetrated the firewall

To be most effective, a firewall ruleset must be augmented with a successful logging feature.

The logging feature documents how the firewall deals with traffic  types. These logs offer insights into, for example, source and  destination IP addresses, protocols, and port numbers.

**When and why firewall logging is useful**

* To see if new firewall rules work well or to debug them if they do not work properly.
* To discover if any malicious activity is occurring within your  network. However, it doesn’t offer the information you need to isolate  the source of the activity.
* If you identify repeated unsuccessful tries to access your firewall  from a single IP address (or from a group of IP addresses), you may wish to create a rule to stop all connections from that IP.
* Outgoing connections derived from internal servers, for example, web servers, may show that someone is using your system as a launch pad.  They could be launching attacks against computers on other networks from your system.

**Linux Firewall Logs**

The Linux kernel has a packet filtering framework called netfilter.  This framework lets you permit, drop, and modify the traffic that comes  in and out of a system. A tool, iptables furthers this functionality  with a firewall, which you can configure using rules. Additional  programs, like fail2ban, also rely on iptables to block attackers.

**How does iptables work?**  
 Iptables is a command-line interface to the packet filtering  capabilities in netfilter. However, we won’t distinguish between  iptables and netfilter for the purposes of this article. To keep things  clear we will refer to the entire concept as iptables.

The packet filtering function offered by iptables is structured as  tables, targets, and chains. Put simply, a table lets you process  packets in a certain way. The filter table is the default table.  
 Chains are connected to these tables. You can monitor traffic at  different points, using these chains. You can see traffic, as it arrives on the network interface or before it is passed over to a process. You  can also add rules to the chains so that they match certain packets, for example, TCP packets going to port 70, and connect it to a target. A  target will determine if the packet should be permitted or blocked.

When a packet enters or exits (according to the chain) iptables  compares it against rules in these chains one at a time. When it  identifies a match, it isolates the target and carries out the required  action. If it doesn’t identify a match with any of the rules it carries  out the action specified by the default policy of the chain. The default policy also acts as a target. By default, all chains have a policy of  permitting packets.

**Working with and interpreting iptable firewall logs**  
 To create firewall logs, the kernel needs to be firewall logging enabled. By default, matched packets are logged as kern.warn (priority 4) messages. You can change the log priority with the --log-level option to -j LOG.

The majority of the IP packet header fields are disclosed when a  packet matches a rule with the LOG target. By default, firewall log  messages are written to /var/log/messages.

**Windows Firewall Logs**

Microsoft Windows has a built-in firewall. The firewall does not log  any traffic, by default. However, you can choose to configure the  firewall to log connections that are permitted and traffic that is  dropped. If you authorize Windows firewall logging, it creates  “pfirewall.log” files in its directory hierarchy. You can see the  Windows firewall log files via Notepad.

**How to enable windows 10 firewall logs**  
 Go to Windows Firewall with Advanced Security. Right-click on Windows Firewall with Advanced Security and click on Properties.

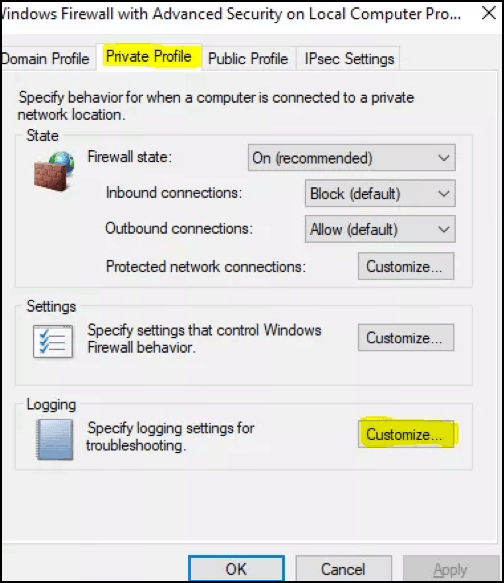


*Figure 1: How to setup logging and tracking:*[*Source*](https://techspeeder.com/2017/07/10/how-to-setup-windows-firewall-logging-and-tracking/)

The Windows Firewall with Advanced Security Properties box should appear.

You can move between Domain, Private, and Public Firewall profiles.  Generally, you should configure the Domain or Private Profile. Let’s see how to create Windows Firewall logging on a Windows Firewall Private  Profile. The steps below will work both for a public profile or a  domain.

Click Private Profile > Logging > Customize



*Figure 2: How to setup logging and tracking:*[*Source*](https://techspeeder.com/2017/07/10/how-to-setup-windows-firewall-logging-and-tracking/)

Go to “Log Dropped Packets” and switch to Yes

Generally, we turn on logging for “Log Dropped Packets” only. We  won’t log successful connections as successful connections generally are not so helpful when resolving problems.

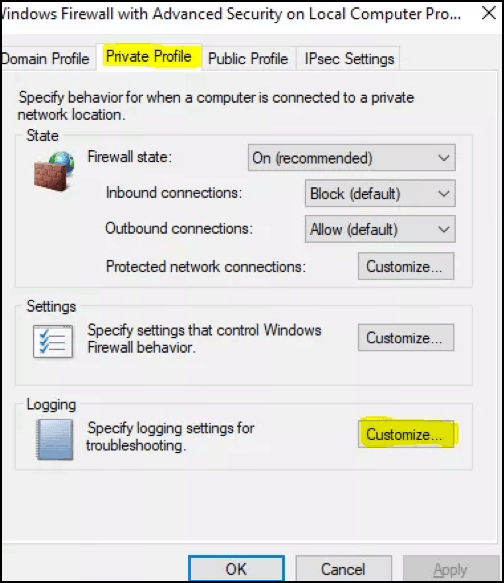
Copy the default path for the log file. ( %systemroot%\system32\LogFiles\Firewall\pfirewall.log ) and then Press OK

Open File Explorer and go to where the Windows Firewall log is kept. ( %systemroot%\system32\LogFiles\Firewall\) . You will see, in the  Firewall folder, a pfirewall.log

Copy the pfirewall.log to your desktop. This will let you open the file with no firewall warnings.

**Interpreting the windows firewall logs**

Your Windows Firewall log will look something like the following:



*Figure 2: Track internet activity with windows firewall log:*[*Source*](https://www.maketecheasier.com/track-internet-activity-windows-firewall-log/)

Here is an analysis of the key aspects of the above log:

1. The time and date of the connection.

2. What became of the connection. “Allow” means the firewall permitted the connection, while “drop” means it has prevented it.

3. The kind of connection, TCP or UDP.

4. The IP of the source of the connection (your PC), the IP of the  destination (your desired recipient, e.g. a webpage), and the port used  on your computer. You can use this to identify any ports that need  opening for software to work. You should also look out for any  suspicious connections, as they may indicate malware.

5. Tells you if this connection was your computer receiving a packet of data or sending one.

**How to Analyze Firewall Logs**

Firewall logging, especially of permitted events, can be helpful for  discovering potential network security threats. An organization  generally places strict protection on assets that should not be freely  accessible. These may include internal corporate networks and the  workstations of employees. Typically, no unmediated inbound connection  to these systems is allowed.

**What to look for when performing firewall log analysis**  
 Once you have gathered the firewall logs and started the process of  analyzing the logs, you can decide what to look for in the log. You  should refrain from only looking for “harmful” events. Your firewall  logs not only help you isolate compromises and incidents, but they can  also help you specify the normal operations of the firewall.

One way to see whether the behavior that has been logged is  suspicious is to see what the normal operations are and then to note the exceptions.

Some events should always cause suspicion and prompt further investigation. They are as follows:

* Authentication permitted
* Traffic dropped
* Firewall stop/start/restart
* Firewall configuration modifications
* Administrator access granted
* Authentication failed
* Administrator session ceased

**Log Analysis and Alerting with Exabeam**

**Log Management and Next Generation SIEMs**  
 Log management is challenging, and it is becoming increasingly so with  the rapid growth of network devices, microservices and cloud services,  endpoints, and the vast increase in data and traffic volumes.

Next-generation Security Information and Event Management (SIEM)  solutions can assist you with your management of security-related log  events and help you learn about events that are relevant to a security  incident.

Features of next-generation SIEMs:

* Security data lake, which can retain unlimited data volumes of historical logs
* User and entity behavioral analytics technology for improved threat detection via behavior analysis.
* Automated incident response capabilities, which provide service  integrations via playbooks for automated investigation, containment, and mitigation of incidents.
* Advanced data exploration features, which can assist security analysts with their threat hunting activity, by examining logs.