Technical Disclosure Commons

Defensive Publications Series

July 2021

NETWORK DATA OBJECTIVIZATION, CLASSIFICATION, VERIFICATION, AND PRIVACY VIA RING-ORIENTED METADATA

Dave Zacks

Nagendra Kumar Nainar

Michele Guel

Carlos M. Pignataro

Tim Szigeti

See next page for additional authors

Inventor(s)
Dave Zacks, Nagendra Kumar Nainar, Michele Guel, Carlos M. Pignataro, Tim Szigeti, and Robert Barton

NETWORK DATA OBJECTIVIZATION, CLASSIFICATION, VERIFICATION, AND PRIVACY VIA RING-ORIENTED METADATA

AUTHORS:
Dave Zacks
Nagendra Kumar Nainar
Michele Guel
Carlos M. Pignataro
Tim Szigeti
Robert Barton

ABSTRACT

Data from network devices is commonly made available without any regard to, or concern, for the ability to provably verify the classification level of the involved data. The owner of a network device frequently will wish to restrict data access, visibility, and processing as a policy action. To address these types of challenges, techniques presented herein support a multi-step approach to addressing the issue of how owners of network device-generated data may share such data with other parties (e.g., a vendor's technical assistance center, partners, etc.) in a controlled way that respects data and other privacy controls and provides verification of the integrity of the data. The presented techniques support, among other things, parsing, objectifying, classifying, verifying, and, optionally, encrypting multiple elements of network device-generated data streams and attaching the output of such a process as verifiable metadata that is associated with the various network data objects thus generated.

DETAILED DESCRIPTION

Today, data from network devices is made available and exported without any regard to, or concern for the ability to provably verify, the classification level of the involved data – such as, for example, privacy levels, the presence of personally identifiable information (PII) in a data stream, data sovereignty restrictions that may be applicable, etc.

Such devices generate different types of data and that data may be consumed by a range of applications, and, in some instance, different groups of engineers, for various reasons. For example, data such as a 'show tech' from a network device, packet captures (e.g., PCAP files) from a device, records encompassing Internet Protocol (IP) traffic flows

that are traversing a device and which are exported to a central location, and logging (e.g., syslog) messages, among other examples, all contain varying amounts of PII data and/or other data which for a variety of reasons (even beyond PII) are collected and exchanged with a vendor's technical assistance center engineers for troubleshooting purposes. On the other hand, streaming telemetry data from the devices may be consumed by different vendor-developed or certified applications, or customer home-grown open-source applications, for analytics, visualization, or optimization purposes. Such network-generated data may comprise varying levels of granular information that could be employed by malicious users to gain business or infrastructure insight (e.g., how a site is connected, the number of internal subnets, etc.) or discover details about potential attack vectors (e.g., allowed ports, enabled features, software version, etc. and the associated vulnerabilities). A data owner may wish to restrict various of the above as a policy action.

However, without the ability to properly segment the data streams, classify the segments, and provide provable integrity for the data segments and elements involved – all capabilities that are lacking today in network devices – no follow-on policies can be safely and comprehensively implemented.

To address these types of challenges, techniques are presented herein that support, among other things, methods to objectize network-generated data streams, classify such data object 'chunks' into varying levels of privacy (based on a ring-oriented scheme), attach metadata to each sub-objectized data block or chunk, verify that the data objects have not been tampered with (either in transit or at rest) thus ensuring data integrity, and finally leverage the above to apply different policies to control, for example, data access, visibility, and processing.

In brief, aspects of the techniques presented herein form the basis for follow-on organizational and supra-organizational policies regarding who can access data – along with options for where, when, why, etc. – thus achieving an organization's goals of controlling access to the network device data that the organization generates and shares. For example, an organization may elect to share network data with a vendor's technical assistance center, or with a partner, for the purposes of troubleshooting.

Various of the techniques presented herein support a method and proposed apparatus for network device data objectivization, classification, and verification, for the

purposes of enabling appropriate data privacy controls, using verified ring-oriented metadata that is attached to the original data stream.

Aspects of the techniques presented herein support a multi-step approach to solving the issue of how owners of network device-generated data may share such data with other parties (e.g., a vendor's technical assistance center, partners, etc.) in a controlled way that respects PII and other privacy controls and provides verification of the integrity of the data.

Considering the above, aspects of the techniques presented herein may be explicated with the help of an illustrative example which captures the techniques' basic approach. Such an illustrative example is presented in the narrative below.

To begin the illustrative example, consider that network data is generated by a network device. This may include, for example, 'show tech' output, 'show run' output, records encompassing IP traffic flows that are traversing a device, logging (e.g., syslog) messages, or other network device-generated data streams.

The generated data streams may be directed from the source device (e.g., a vendor-supplied switch, router, etc.) into a containerized application that is running on the device (e.g., running within a 'guest shell' under a device's operating system (OS)). A containerized application is used in the instant example as just one possible approach (e.g., providing a faster time-to-market for aspects of the techniques presented herein by avoiding the need to code changes to the OS base on the device to implement the below described functionality). However, conceptually this would be identical if the base OS on the device also directly provided the below described functionality.

The containerized application (which for simplicity of exposition may be referred to as a Network Data Objectivization and Classification (or NDOC) engine) performs a number of functions on the network data stream that is sent to the NDOC engine. Those functions are described in the narrative below.

The NDOC engine parses an incoming data stream and chunks the data into various 'objects,' with each object corresponding to an element over which differentiated control may later be desired. For example, in a 'show run' output objects may include the Authentication, Authorization and Accounting (AAA) portion of the configuration, the routing portion, the various interface configurations, etc. It is important to note that objects can be, and most often would be, granular all the way down to individual data items such

as IP addresses, port numbers, AS designations, etc. Objectification is not limited to the section level. This allows for, as an example, an interface-level configuration to later be reviewed but to retain control over viewing or use of the IP addresses involved.

The NDOC engine assigns each object that is parsed within the network data stream to a classification level based on a defined ring structure, where Ring 0 is the most private and leading out to Ring 'N' being the least private. A minimum of three ring levels would typically be defined for network data streams (supporting highest privacy, medium privacy, and lowest privacy), but additional rings may be defined and used, if needed. The decision as to which elements within an incoming network data set would be attached to which privacy or security classification level is driven by a rules-based engine that is resident within the NDOC. A network device administrator is able to override such a classification level and assign, either temporarily or permanently, a different level if desired. For example, if an administrator determined that for some reason certain data elements that normally reside at Ring 1 should be reclassified to Ring 0, at any point in time they could make such a designation.

The NDOC engine leverages an on-device ‘trusted anchor’ module (TaM) (e.g., an on-device hardware-anchored immutable basis for a trust chain) to digitally sign each network data object that is generated. This accomplishes two things. First, it guarantees that the network data in fact originated from the origin device that it claims to have originated from. Second, it verifies that this network data object, when it is later reviewed, has not been tampered with, either while it was in-flight or while it was at-rest. Additionally, the NDOC may attach other metadata to the objects to provide a contextual description of where the data originated. For example, this may include an identifier in the networking device describing the country, state, province, federal agency, etc. where the device is used.

The NDOC engine attaches the metadata (e.g., the classification ring level and the associated network data object, the ring level digital signature, the contextual information, etc.) to each and every network data object element. Optionally, the NDOC engine can encrypt the resulting fleshed-out network object to provide data obfuscation, if desired. Note that some network data objects so produced could be encrypted, while others might not be encrypted. If encryption is used, the encryption keys may be controlled by the

network data source (typically the owner of the network device) and may be provided to the remote possible end-user of the network data object (for example, to a vendor's technical assistance center or to a partner) via an appropriate key distribution method. In this way, the network device owner (typically the customer) can then revoke the keys if needed, disallowing access to the data (even when the data is held remotely) by denying the ability to decrypt it.

Finally, to complete the illustrative example, the NDOC engine exports from the network device the resulting objectivized, classified, and verified (and, optionally, fully or partially encrypted) data elements for off-box processing and use.

Off-box devices may now use the object-based, classified, and verified series of data elements as they see fit. For example, policies may be put into place such that, for example, a vendor's technical assistance center is able to view and use all Ring 1+ data but Ring 0 data (such as, for example, passwords, designated IP addresses, etc.) would be restricted, an analytics engine processing the network device data might only be allowed Ring 2+ access, and a trusted network administrator might be allowed Ring 0 access.

It is important to note that while rings often may be concentric (i.e., having Ring 0 access implies having access to all of the rings above it, having Ring 2 access implies denial of the rings below it, etc.), another (e.g., more flexible) access model could also be instantiated that could allow differentiated access on a per-user or per-group basis (i.e., to override or constrain the ring-based access method, as needed or desired).

A customer may also stipulate policies that examine other contextual elements of the metadata in the data objects, such as preventing certain types of data from being stored outside of a data sovereignty region (e.g., a U.S. federal agency router producing data could have identifiers preventing a vendor's technical assistance center from storing those files outside of the United States).

Figure 1, below, depicts how network data may be objectized with different metadata policies to control the consumption by different parties according to aspects of the techniques presented herein.

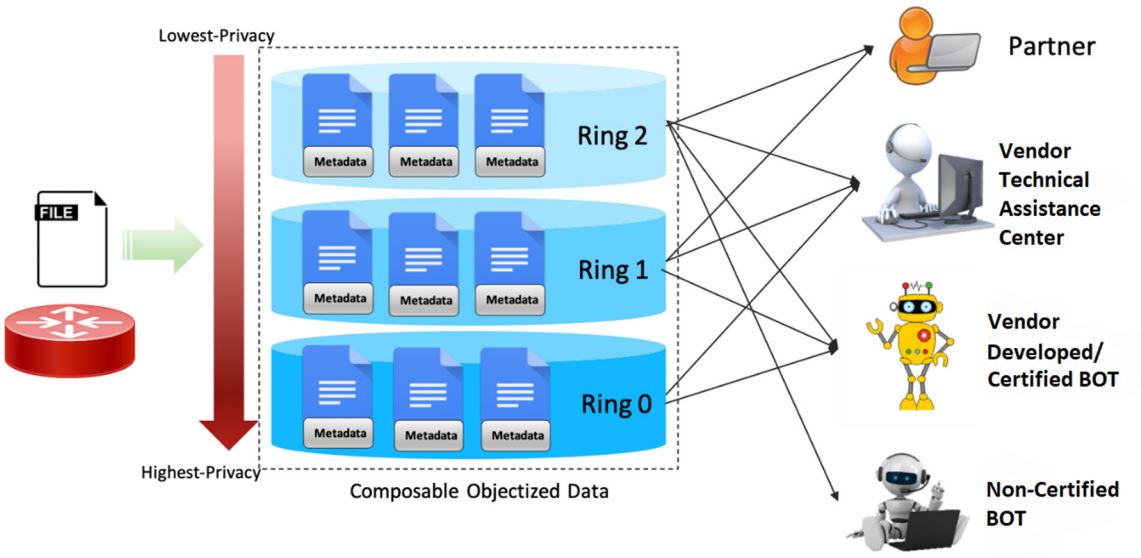

Figure 1: Illustrative Ring-based Network Data Objectivization

Figure 2, below, presents a very simple example of how a 'show output' may be objectivized with additional metadata according to aspects of the techniques presented herein. For brevity, just the security level is highlighted and a JavaScript Object Notation (JSON) format is employed. However, such brevity does not imply any implementation decisions or limitations.

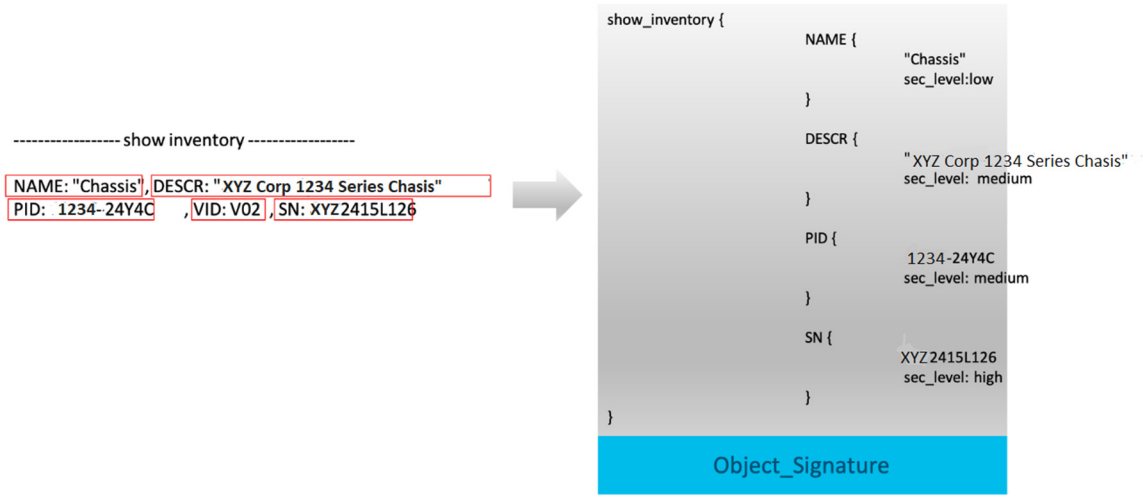

Figure 2: Exemplary Network Data Objectivization

The example from Figure 2, above, employs an inventory model and depicts the association of different security levels for different data. The output could always be generated as a binary file and so depending upon the security clearance of the user who is reading the file the relevant objects may be composed together. For example, a user with maximum security clearance could see the serial number (i.e., SN) in the above output while a user from a partner may just see the chassis name (i.e., NAME).

To briefly summarize the above discussion, the techniques presented herein support a novel method and propose an apparatus for parsing, objectifying, classifying, verifying, and optionally encrypting multiple elements of network device-generated data streams and attaching the output of such a process as verifiable metadata that is associated with the various network data objects so generated. Such capabilities do not exist in network devices, or in the industry, today.

While some of the elements that are found under the techniques presented herein (such as the different data elements involved and an on-device 'trusted anchor' module in network devices) have been available for some time, no current solution exists as a whole according to the techniques presented herein – i.e., network device-generated data being objectified, classified, and verified with such processing then used to tag verifiable metadata onto the resulting network data objects for the purposes of controlling off-box network data element usage, distribution, and policy-based controls.

In particular, aspects of the techniques presented herein focus on chunking the desired data into objects, attaching metadata tags that allow each chunk of data to be associated with a given level of data sovereignty, and ensuring that the integrity of the data chunks is not tampered with – for all aspects of network device-generated data.

This includes not just configuration files from devices but also data sets that are generated by network devices such as records encompassing IP traffic flows that are traversing a device, logging (e.g., syslog) messages, or other network device-generated data streams, all of which are provided today in a 'flatter' format without any associated metadata tags or classification information that may be used to support data sovereignty efforts. In contrast, techniques herein provide for the ability to segment, metadata-label, and secure all aspects of network-device-generated data with an eye to using these chunking, classification, and signing capabilities in support of data sovereignty efforts.

Additionally, the techniques presented herein attach a digital signature to any and all such device-generated data and metadata, anchored into the immutable on-device 'trusted anchor' module within the device, to ensure that the data and metadata has not been tampered with in transit or at rest. Such an approach is important considering that policies surrounding access to the data will be based on the attached metadata classifications and the data sovereignty levels that they imply. This is an important consideration given the absence of such capabilities in today's 'flat' and non-chunked device-generated data sets.

It is important to note that a customer cannot control how their data is accessed or handled by partners, a vendor's technical assistance center, or others once the data is out of their device or domain. Consequently, it is the application of the chunking, metadata attachment, and digital verification capabilities to all aspects of device-generated data that is a key to the techniques presented herein.

Due to the need to apply complex data sovereignty rules to network device-generated data, the techniques presented herein add an important new capability to a vendor's network devices and can serve as a critical differentiator between the vendor and other vendors' devices when it comes to the critical area of data sovereignty and control for all aspects of network device-generated data streams.

As the metadata that is attached to each chunk of network-device data is key – that metadata can be used subsequently in many different ways for data sovereignty purposes – it should be noted that under the flexible techniques presented herein that metadata may be generated in several unique ways. For example, the metadata that is attached to a chunk of network device-generated data may be based on a combination of not only the ring level but also the context of where a device is located (e.g., is it a core device vs. an access device vs. a firewall, etc.). In this way, the ring level for the metadata may be dynamically defined based on the device's role, position within the network, or other factors. Such a capability does not exist today. Additionally, concerning the use of the metadata, rather than having a static set of rules that define what data and data types are associated to which ring level as a classifier, at each ring level a machine learning (ML) classifier may be used that takes a series of inputs from the associated data stream and subsequently determines the best ring

level for all or a portion of the involved network device-generated data. Such a capability does not exist in the industry today.

Use of the techniques presented herein offers clear visibility. For example, use may be concluded if it is observed that network device data that is being generated (either directly or via an adjunct element such as an on-box or off-box container) contains a segmented, objectivized version of the network device data, tagged with metadata concerning classification levels, and verified by a digital signature or similar mechanism to avoid data tampering.

The techniques presented herein will be extremely useful for any entity wishing to, or needing to, control access to their network device generated data once that data leaves the network device. The classification (using a ring-based metadata system) provides a flexible method of network device data classification and integrity verification, upon which a separate but associated policy infrastructure for appropriate data access may be created. A customer subject to data sharing restrictions (organizational, geographical, or otherwise) will find the techniques presented herein particularly useful.

In summary, techniques have been presented herein that support a multi-step approach to solving the issue of how owners of network device-generated data may share such data with other parties (e.g., a vendor's technical assistance center, partners, etc.) in a controlled way that respects data and other privacy controls and provides verification of the integrity of the data. The presented techniques support, among other things, parsing, objectifying, classifying, verifying, and optionally encrypting multiple elements of network device-generated data streams and attaching the output of such a process as verifiable metadata that is associated with the various network data objects thus generated.