January 2024

NETWORK TOPOLOGY RECONSTRUCTION FROM OPERATIONAL LOGS

Dmitry Goloubew

Don Allen

NETWORK TOPOLOGY RECONSTRUCTION FROM OPERATIONAL LOGS

AUTHORS:
Dmitry Goloubew
Don Allen

ABSTRACT

Techniques are presented herein that support a new, low-overhead network topology discovery mechanism that is based on data from existing operational log sources such as, for example, syslog entries. Such an approach lowers the operational cost of any solution that requires knowledge of a network's topology. Additionally, such an approach allows for the ongoing rediscovery of a network's topology to track over time the documented or undocumented evolution of the same.

DETAILED DESCRIPTION

Many network-related diagnostic and investigative activities rely on an accurate knowledge of a network's topology. Examples of such activities include, among other things, security exposure analysis, resilience evaluation, traffic pattern analysis, maintenance scheduling, change impact analysis, etc.

Discovering and then maintaining an accurate network topology typically entails the collection and the subsequent analysis of specialized data. Importantly, the availability of that data often requires additional network configuration activity and data collection, typically resulting in an additional operational overhead on the involved networking devices. Further, a networking environment may comprise multiple layers and features, each of which may have its own notion of topology.

Techniques are presented herein that address the above-described challenge. Aspects of the presented techniques encompass analyzing operational logging information (as found in, for example, syslog entries) from a multitude of devices that form, and which are connected to, a network.

Such an approach yields a number of advantages over existing methods, including the discovery of a network's topology not requiring any additional data collection; no additional overhead being imposed on the network in connection with the collection of

available information; such a low-overhead discovery method allowing for the continuous maintenance over time of a network's topology; and such a universal approach allowing for the discovery at multiple levels (including at a layer 1 physical connection, at a layer 2 forwarding level, at a layer 3 routing level, etc.) and in the case where overlay features (such as tunnels) are present in a network.

Under the presented techniques, the discovery of a network's topology involves aggregating operational data. While that data includes, at a minimum, syslog entries, other data sources (such as traces, etc.) may optionally be included to enable the discovery of additional topological layers.

It is important to note that syslog messages are unstructured data having, among other things, different priority levels. A network's topology may be discovered from such data by identifying events that are reported at roughly the same time by both (or rather by all, in the case of a multi-access environment) of the sides of a connection. Examples of the same may include link and port state events, spanning tree state events, routing protocol state events, etc. Additional examples may come from proprietary solutions wherein the messages may be internal to an implementation.

It is also important to note that the content of a message is not used in the basic implementation of the presented techniques. However, message content may be used in an extended implementation to optimize accuracy at the cost of running time or dependencies (for example, a very strict implementation may dictate what messages could ever be related).

Additionally, while a basic implementation may not consider syslog levels or priorities, if it is determined that there may be useful heuristics within the same that information may be included in an extended implementation to, for example, raise the accuracy of discovery.

Further, a production-grade implementation of the presented techniques may also discern different types of connections. For example, a spanning tree state-derived connection implies physical connectivity while a Border Gateway Protocol (BGP) peer-derived connection implies logical connectivity and therefore may be used differently. Under such an approach, the product of a discovery process may encompass a list of edges,

with a type for each edge, which supports the ability to identify different topologies (such as physical, logical, routing, etc.).

As noted previously, under the presented techniques the discovery of a network's topology involves aggregating operational data. The specific data elements that may be aggregated include host information and time information, and optionally facility information.

Time information (e.g., a timestamp) allows for a correlation to be performed within a specified window around the time that a log message was produced. Such an approach relies on a synchronization of the clocks on the different network devices. However, if such a synchronization is not possible or available then the presented techniques are still usable provided that a precalculated correction factor is applied to each source device.

Under the presented techniques, the source of a message is also considered. For example, messages from different devices need to correlate in time in order for the techniques to detect a connection. A common example would be an interface link going down where there will be a message from both ends of the link. Additionally, routing protocol adjacencies will also produce similar correlations, and protocols such as the Bidirectional Forwarding Detection (BFD) protocol will also produce working correlations.

However, the hostname of an adjacent device does not necessarily need to appear in a message. For an accurate topology recovery, it is important that each device be correctly identified. For example, if a particular device sources its messages with an Internet Protocol (IP) address and a hostname, it is important to ensure that both of those values are treated as the same device, which may be achieved through a pre-processing step. During a correlation process, the simplest case may develop an association between hosts over time, but for further accuracy another implementation may subdivide the information and correlate mnemonics within each host.

The presented techniques support a number of hyperparameters, including correlation window size, strength, etc.

Correlation relates the total number of messages to just the interesting messages. In principle, such an approach may work even with a very small number of messages (e.g., ten messages) from each pair of connected devices, but to make the inferred topology more

accurate a practical number should be approximately a week's worth of messages. Of course, such a number will vary from network to network as the types of messages relate more to activity across the network rather than to time. In times of transitions (such as, for example, when an entire network is coming up from a power outage or following maintenance, and all of the devices are loading and synchronizing) a day's worth of logs may be sufficient to produce a reasonably accurate topology.

While it is possible to make the presented techniques work from a single pair of messages (by using a narrow window of correlation and low threshold hyperparameters), such an approach is unlikely to produce a very accurate topology. From a practicality perspective, one would generally want to choose hyperparameters such that a recovered topology is as accurate as possible using the smallest number of messages.

For example, since the above-described approach is based on correlations in time, and thus is probabilistic in nature, accuracy is dependent upon carefully selected hyperparameter values. If the selected correlation window is too large, several events from different sources may 'alias' together and may lead to the discovery of spurious connections (i.e., the connections a ←→ b and b ←→ c may yield a phantom connection a ←→ c). In applications where the cost of an error is high, hyperparameter tuning and convergence detection are highly recommended.

As noted previously, one of the advantages of the presented techniques (in their least restrictive form) is that they do not specifically look at the content of a message. As a result, the techniques can easily support a multivendor solution (where, for example, one side speaks a one 'syslog language' and another side speaks a second 'syslog language'). For example, in the following log entries:

```
router1, 01:01:01, line protocol down E1/1
router2, 01:01:01, interface down G23
```

despite the fact that router1 and router2 describe a link down event through different messages, over time, and through correlation, the presented techniques can discover a strong association between the devices and use that information to create the edges of a graph.

As alluded to previously, a principal advantage of the presented techniques is that in practically every network the above-described information is already collected as part

of normal operational procedures. Using that information, the techniques support the discovery of a network's topology through an analyses of a correlation matrix that is created by using a sliding window to find time-based Pearson correlations between hosts or between facilities. The discovered correlation coefficients that exceed a certain strength may be used to create the edges of a network graph, with devices representing the nodes of that graph.

Regarding correlation strength, a particular implementation may select a cutoff threshold within which the correlation is considered representative of a connection. This may be done with a simple operator-established threshold or, alternatively, an implementation may choose to derive a threshold automatically through, for example, elbow detection and analysis or some other method. If such correlations are performed at a host, mnemonic, and facility level, then the correlations are likely to be more stable, but the data will exhibit a higher degree of sparseness at the mnemonic and facility level. An alternative method would encompass correlating at the host and address level (depending on the noisiness of the logs) which may require less time to recover the actual topology but at the cost of reduced correlations (due to independent messages also appearing in the logs and diluting the correlation).

Figure 1, below, depicts elements of an exemplary device correlation that is possible under the presented techniques, and which is reflective of the above discussion.

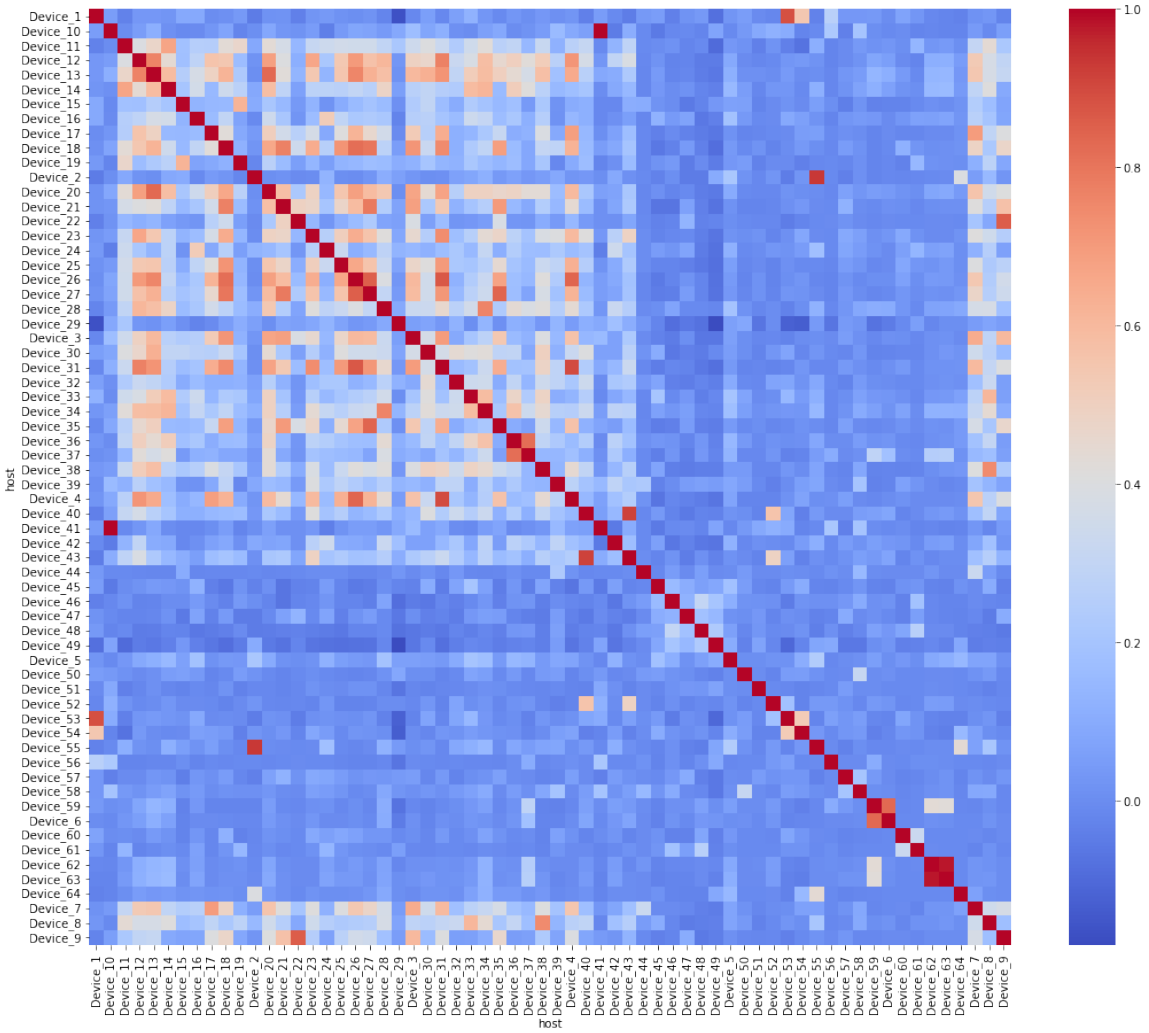

Figure 1: Exemplary Device Correlation

The exemplary device correlation that is shown in Figure 1, above, was developed by applying the presented techniques using three log entries (i.e., timestamp, facility, and source) while ignoring the content of a message.

Under the presented techniques, a default log setting will typically be sufficient to discover a network's topology; in other words, special logging should not be required. This is due to the techniques focusing principally on the amount of collected data based on time. In general, a collection period of one or more weeks should be sufficient for most networks.

As noted previously, the presented techniques support a number of hyperparameters, in particular a correlation window size, a correlation strength, and an observation duration. Additionally, a production implementation of the techniques may

choose to allow different correlation methods (e.g., time and host; time, host, and mnemonic; etc.) in order to provide a user with sufficient flexibility to adjust their operation to a given network.

It is important to note that false positives are possible if the above-described hyperparameters are not correctly chosen. For example, if a correlation window is too large then aliasing (where events that happen at somewhat different times are essentially considered to happen at the same time) may occur.

Figure 2, below, presents elements of an exemplary network graph for the discovered device correlation from Figure 1, above.

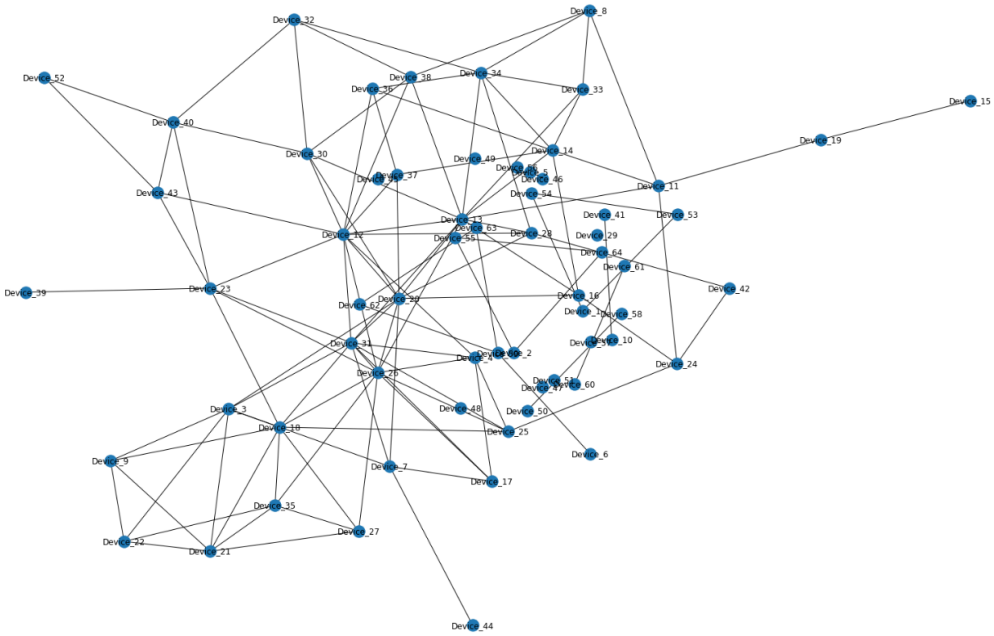

Figure 2: Exemplary Network Graph

The network graph of Figure 2, above, identifies the network connectivity between the indicated nodes and provides a reasonable representation of the network's topology.

As described above, a significant advantage of the novel approach under the presented techniques is that no additional device data (outside of the information that is contained within syslog files) is required. Since syslog files, and the information that they contain, are the lifeblood of a network and its operations, the ready availability of that information when combined with the presented techniques produces a low cost, low entry, and low risk approach to producing the network topologies that are required to troubleshoot

and manage networks. A further advantage of the techniques is that they are, at their core, friendly to multi-vendor and multi-operating system (OS) environments and not based on any fixed or distilled knowledge of any kind.

The presented techniques may be employed in any number of use cases that would benefit from the low-cost discovery of a network's topology to provide additional context to diagnostic, investigative, analytic, etc. activities. Many customers (rightly or wrongly) still are very careful about imposing any additional overhead on their network devices. This means that while many of a network equipment vendor's solutions have a way to discover a topology, that action is performed as a deliberate act with (real or imagined) impact and overhead. As a result, one often ends up without a topology or with a manually drawn, possibly outdated or inaccurate, topology. In contrast, every serious network operation collects syslog entries, and, through the presented techniques, a topology can be quickly discovered without any added overhead or data requests and the associated risks.

In summary, techniques have been presented herein that support a new, low-overhead network topology discovery mechanism that is based on data from existing operational log sources such as, for example, syslog entries. Such an approach lowers the operational cost of any solution that requires knowledge of a network's topology. Additionally, such an approach allows for the ongoing rediscovery of a network's topology to track over time the documented or undocumented evolution of the same.