February 26, 2019

ENHANCED DYING GASP IN NETWORK NODES FOR INTUITIVE MONITORING OF COMPLEX NETWORKS

Karthik Babu Harichandra Babu

Manigandan B

Ananthakrishnan Rajamani

ENHANCED DYING GASP IN NETWORK NODES FOR INTUITIVE MONITORING OF COMPLEX NETWORKS

AUTHORS:
Karthik Babu Harichandra Babu
Manigandan B
Ananthakrishnan Rajamani

ABSTRACT

Techniques are provided herein for fixing existing Operations, Administration, and Maintenance (OAM) Dying Gasp (DG) mechanisms and providing guaranteed delivery of the DG packet to the remote/peer node.

DETAILED DESCRIPTION

Most service provider network utilities are managed by third party vendors (service providers) and are governed by contracts referred to as service level agreements (SLAs). For example, an SLA is a contract defining how a vendor is to manage/maintain a service provider network utility. In situations in which an SLA is not met, the service provider can charge a penalty based on the utility issue(s) affecting the network. In certain circumstances, in may be beneficial to enable service providers to clearly isolate failures in the network that are due to the utility, or due to system failures. Such capabilities would enable the service provider to charge the third party vendors for not abiding by the applicable SLA(s). Accordingly, presented herein are techniques that provide service providers with the ability to determine the exact reason for a failure.

For example, if there is a power failure, it is desirable for the service provider to determine whether the failure is a result of a utility failure, and not the system. This information provides the service provider with a clear understanding of the underlying issues and, accordingly, enables the service provider to hold the third party vendors accountable for violations of the SLA(s). That is, the techniques presented herein enable the service provider to determine exactly what caused the outage and who is responsible for the outage. Additionally, the techniques presented herein describe actions, automation, and analytics that may further improve the service provider customer experience in a unified/integrated solution.

The techniques presented herein may be used in the context of two mechanisms provided to service providers for issue troubleshooting, namely: (1) Onboard Failure Logging (OBFL), and (2) Dying Gasp (DG). Use of the techniques presented herein with each of these mechanisms is described in greater detail. However, it also to be appreciated that there may be other options for the vendor to implement the underlying techniques presented herein to improve efficiency when dealing with network issues and third party vendors.

Referring first to OBFL, this is a mechanism for applications to store hardware troubleshooting data into non-volatile memories for diagnostics and repairs. In traditional arrangements, this is a tedious process that involves uninstalling the hardware and then performing the diagnostics.

DG refers to a mechanism that provides troubleshooting functionality. More specifically, a DG event is a message (or signal) sent by the host network equipment/router to remote logging devices using, for example, the Simple Network Management Protocol (SNMP), syslog, or Ethernet Operations Administration and Maintenance (OAM) to report any abrupt loss of power to the host network equipment. In certain arrangements, a DG event (message) is triggered when certain unrecoverable/recoverable conditions occur, such as: (1) complete power failure; or (2) removal of the input power supplies cable. However, it is to be appreciated that that the specific conditions triggering a DG event is vendor specific.

In general, an Ethernet OAM notification about the condition may be sent immediately, along with the SNMP DG trap. For example, the message may convey the following information in its syslog: “Existing message provides failure reason and interface information.” However, platforms using the DG feature do not provide guaranteed delivery of this message due to, for example, limitations inherent in those platforms. The aforementioned message also does not convey additional details about the dying remote node, which in turn limits the ability to quickly find the power/link down information as in a larger network.

The techniques described herein address the drawbacks in a DG event and provide several benefits. First, the techniques presented herein provide guaranteed delivery of critical DG events/messages across all platforms. Second, service provider customer add-

ons (e.g., host name and IP[v4|v6]address of the interface) are provided to bolster the troubleshooting. Third, a hashing algorithm (cryptographic hash) is introduced to store the DG parameters. A cryptographic hash is a one-way function that is deterministic, fast to compute, resistant to pre-image and second-pre-image attacks, and collision resistant. This comprises one-way functions that produce a "fingerprint". Essentially, hashing algorithms map a value with many bits to a value with a smaller number of bits (128 bits in the case of MD5) in such a way that collisions are as rare as possible. This is highly useful for platforms which have a space constraint to send out the DG message.

Fourth, the techniques presented herein assist in the avoidance of man-in-the-middle attacks. Fifth, the techniques presented herein provide monitoring of the devices that is enhanced using an intuitive User Interface (UI) layer composed of a mix of voice/visual and touch interface, such as speech Interactive Voice Response (IVR) based technology.

Current platforms are unable to log reasons for failures. The few implementations across the industry for product decoding are very complex and must be performed at the vendor premises. Moreover, the DG interval is minimal in many platforms, which makes it merely a “best effort” delivery, rather than a guaranteed delivery. In general, no clear distinctions exist and critical messages may be hacked because the raw data format can be used. Although monitoring can be performed through servers, this causes scalability issues and is insufficient to achieve the needs of service providers. As a result, the conventional DG messages do not convey the complete context of an issue so as to enable immediate and complete troubleshooting.

Figure 1, below, illustrates an example model of the existing DG mechanism.

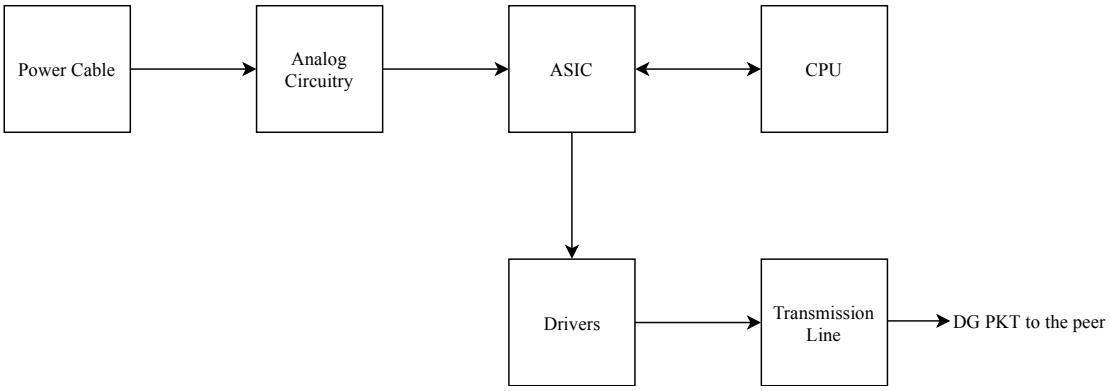

Figure 1

Figure 2, below, illustrates an example model for sending a DG when a power outage occurs.

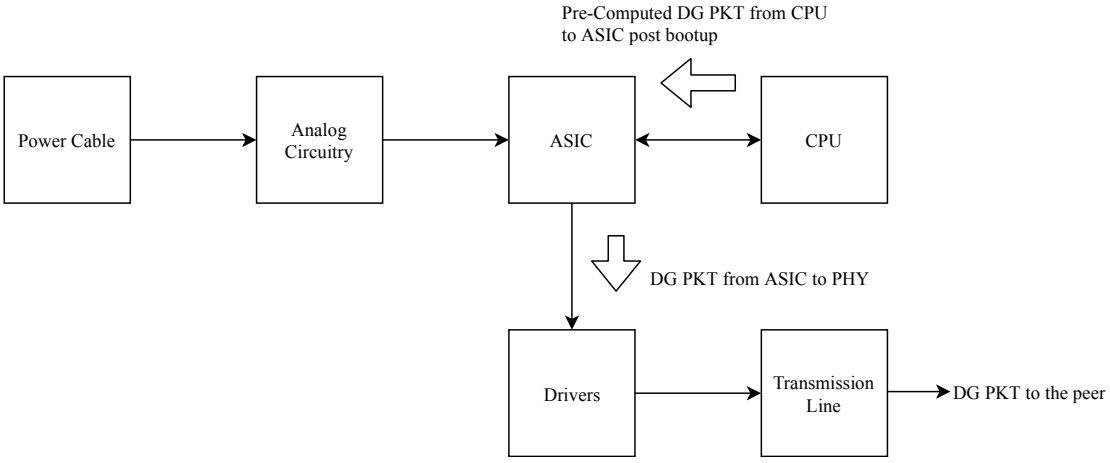

Figure 2

When the system boots up, a Central Processing Unit (CPU) may statically craft a packet (DG) and store it in the unused portion of the Application Specific Integrated Circuit (ASIC) memory. When the analog circuity detects the power failure, the ASIC may directly insert the packet into the transmission line using driver support, which overcomes the need for the CPU to dynamically generate a packet and send it out on a transmission line before the network equipment dies.

The DG packet may be implemented successfully in all platforms and promise a guaranteed delivery, and may also be enhanced. Additionally, the network may require the dying remote node information as soon as possible to generate power/link down information in a central server for quick troubleshooting or damage control. For example, after a power failure, a device has 3-14 milliseconds to send the DG message out, providing as much information as possible to the local peer to enable quick action. Appending a remote node hostname and Internet Protocol (IP) address in the link OAM packet may provide further useful information to the service provider. Additionally, the dying host IP address/host name may be provided in the link OAM packet generated in its priority Type-Length-Value (TLV). It may be displayed in the local peer syslog/DG using the DG message.

The existing broken DG feature does not indicate that the neighbor went down. Moreover, it fails to detect which neighbor went down by clearly mentioning the

corresponding host name and IP address. Instead, it merely specifies the reason for the failure and the interface. Accordingly, as described herein, the remote IP address and host name may be included.

In the access implementation, if there are several neighbors connected across the same broadcast network, the current DG message may not distinguish which IP address/host device went down. Additionally, some platforms may have space and time constraints that limit the ability to add new fields in a DG message that is sent out when a power failure occurs. To address this issue, hashing techniques may be used to reduce a large number of bits to a smaller number of bits and to enable platforms to transmit easily without any constraints. In general, a hash is a hexadecimal string of several characters. Hashing is also a unidirectional process, meaning that it is not possible to work backwards to obtain the original data. The use of hashing may also improve the security of transmissions of critical messages in a complex and confidential network. Hash maps are fast (e.g., O(1)) in the best case.

Existing protocols (e.g., Link Layer Discovery Protocol (LLDP)) are used without fail across all service provider customers, and may be employed and used between peers. Their payload information, which includes the IPv4|v6 address, hostname, and interface on which they communicate, may be hashed and archived under the ASIC/driver/interface code (in high end platforms) in unused memory. The entire packet may be built, while the system only sends the hash of the payload. The receiving end examines the hash information from its table, reconstructs the complete packet, and obtains the entire message. This may be similar to a control-plane interaction during boot-up where later the hash/ID/message is used to reconstruct the whole packet. The mechanisms described herein may enable the system to ignore or circumvent the limitations on large packets to be sent on transmission media when there is very little time to construct the entire DG message from the beginning. Often, the shutdown occurs with relatively small gasp time available for platforms (e.g., 1-12ms), but the hashing messaging described herein may still be implemented with a smaller frame size. Obtaining logic from the hash may be possible (reversible) at the receiving peer/end/node.

The techniques described herein address shortcomings of existing models relating to frame size constraints, time constraints (addressed by implementing a hash table with

the necessary details), gasp interval supported per platform port density basis, and hashing techniques (which can even store the DG packet headers, thereby allowing less transmission time and hence supporting a lower gasp interval).

Figure 3 below illustrates an example implementation of the techniques presented herein.

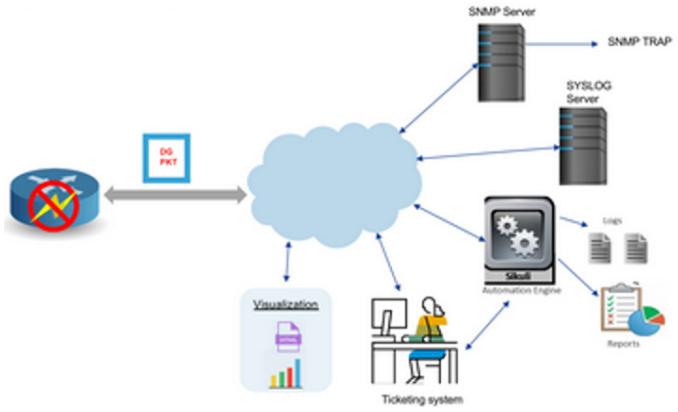

Figure 3

In general, a DG message may be monitored in many ways, such as at via the SNMP server to SNMP trap, syslog server, splunk (a syslog analyzing engine in which all systems utilize one syslog automation mechanism for the reporting UI), automated alert (e.g., automation engines that can parse the format provided and construct a report), Graphical User Interface (GUI) / visualization, and intuitive management of networks with voice/touch User Experience (UX) (speech or IVR based). Generating a DG message in the format described herein may enable automation and, in certain examples, syslog monitoring may effectively take the necessary next steps to avoid the issue firsthand. The same message may be formed as a SNMP trap which may be effective in troubleshooting the issues.

There are implementation differences between normal SOS messages and the messages described herein. For example, the CPU need not become involved because the ASIC may handle sending the DG message. Previously, the CPU may have generated and transmitted the packet. Additionally, the dying device may send the link OAM SNMP trap directly from the hardware to the Network Management System (NMS), rather than sending a link OAM DG Protocol Data Unit (PDU) and relying on the peer to generate the trap. The direct SNMP DG trap generation on the dying router offers many improvements.

In summary, presented herein are techniques that provide a number of benefits to service providers. These benefits include guaranteed delivery of critical DG messages, whereas the existing model is based on best efforts. The associated implementations may rectify all the previous measures and provide a guaranteed delivery. Add-ons such as host name and IP[v4|v6] address of the interface may be available to bolster troubleshooting. Additionally, a hashing algorithm is introduced to store the DG parameters, thereby strengthening the security of the network.