SCTE ISBE CABLE-TEC EXPO '16
SEPTEMBER 26-29 PHILADELPHIA

Society of Cable Telecommunications Engineers
International Society of Broadband Experts

Autonomic Networking Infrastructure

The Foundation For Next-Generation Operations, Maintenance And Automated, Software Defined Networks.

A Technical Paper Prepared for SCTE/ISBE by

Toerless Eckert
Principal Engineer
Cisco Systems Inc.
170 W. Tasman Drive, San Jose, CA 95134
408 902 2043
Eckert@cisco.com

Table of Contents

Introduction

OAM and Service Automation in the Data Center (DC) are quickly evolving and driving technology evolution and standards in SDN and NFV. Trying to apply these methods to wide-area, metro or customer premises networks introduces a range of unique challenges.

OAM and Service Automation for wide-area, metro or customer premises networks, consists of a highly fragmented complex set of technologies and common practices. This includes areas such as Zero-Touch-Deployment for devices (ZTD), provisioning, security mechanisms and reliable OAM infrastructure - especially in complex topologies such as a multi-layer subtended ring structure. Vendor or platform type specific solutions are common.

This presentation discusses an evolving architectural approach to address these challenges called "autonomic networking" (AN). It is currently being standardized via the ANIMA working group in IETF. AN intends to provide a common software infrastructure across network platforms in all market segments.

Security operations for autonomic network is done via zero-touch public key certificate management, simplifying this otherwise often most complex aspect of network management.

The security design relies on and enables a secure, zero-touch built, in-band virtual management network. This so-called "Autonomic Control Plane" (ACP) is indestructible by operator configuration or SDN application, including mistakes or intentional changes to connectivity/services/security.

The ACP provides secure IPv6 connectivity and service discovery not only for such OAM/SDN operations, but also for future intelligent distributed autonomic agents. Proactive fault management via an architecture of such intelligent distributed agents running in self-managing network elements and a hierarchy of self-managing NMS, is an important candidate use case for this AN infrastructure.

Content

1. Background: The Problems

In this first section, we investigate the evolving and current state of the network that was seen as the problems which caused us to work on autonomic networking.

1.1. Evolution and current state of transport for OAM in professionally managed networks

Connectivity and transport architectures for remote network equipment deployment and remote network management (OAM) have gone through multiple iterations in the last decades.

1.1.1. The Dark Ages

Originally, different type of network equipment had no other common management interface than serial port "console" connections: PBX, TDM, Frame-Relay/ATM modems, crypto devices and so on. The

resulting architecture for remote OAM therefore consisted for a long time of modem (banks) into these console port, making the PSTN or private telephony networks likely the first generation of completely separate management networks. X.25 networks with (reverse) PADs became popular in the 80's and early 90's (at least in Europe).

1.1.2. Data Communication Networks (DCN)

With the proliferation of TCP/IP technologies in the 90's, it became clear that the IP network was a better common transport infrastructure than telephony or X.25 networks, even if it was by itself built potentially on top of much of the same underlying layer 2 technologies in those days (e.g.: ATM, Frame-Relay) Out of this, the ITU-T architecture evolved in the early 2000's: G.7712/Y.1703, called the "Data Communication Network". At the transport layer, this is simply an IP network, but with the ability of all Telco equipment to be managed via TCP/IP (instead of only serial ports), a rich framework of management protocols could be adopted that had already developed in pure TCP/IP network - for example SNMP. Bootstrapping equipment as well as low-level diagnostics and device manageability often still required serial ports though, so terminal servers were also abundant in those DCN.

DCN where and are primarily constrained to large service providers because while they do provide a great degree of reliability, they also are very expensive both in CAPEX and OPEX, and they are often seen as difficult to use by operators.

1.1.3. TCP/IP networks: In-band OAM

Meanwhile, actual TCP networks in most market segments were most often built with only "in-band management": The actual TCP/IP network and transport services for customers were also used to carry OAM traffic. For probing the correct functioning of subscriber services, doing this probing in-band is often the most accurate form of observation, but for many other OAM operations this approach is detrimental. Any failures in subscriber transport service configuration (e.g. routing) would also make OAM operations fail. Likewise, access to the OAM plane needs to be protected from attacks by subscribers. When network paths became overloaded by subscriber traffic, OAM operations would fail.

Over time TCP/IP network operators developed a wide variety of methods to create forms of isolation between OAM and subscriber traffic. Address management was probably one of the first and still used solutions: Access Control Lists (ACLs) were defined to allow access from/to OAM addresses/ports of network equipment only from/to NOC addresses/ports. The most common scheme is what has been called "clam shell security". Service provider networks use ACLs on interfaces connecting to subscriber equipment to filter out traffic from/to OAM addresses/ports. If set up correctly, this approach creates a fairly hard shell of external protection, but once passed, there is no further security. Alas, maintaining these lists so that they do not permit any undesired traffic but also do not deny any desired OAM traffic is not free. Especially not when addressing in networks evolves. Foremost, security filtering does not allow to keep OAM services up and running when there is a misconfiguration or other issues in the subscriber transport service itself.

1.1.4. Homegrown OAM security and isolation

To solve this issue in TCP/IP networks, isolation between OAM and subscriber traffic is being set up with a variety of mechanisms, typically utilizing exactly the type of isolation that the network may already use between different types of subscribers. Early enterprise networks were built with mostly L2 services (VLANs) per subscriber, so another VLAN was assigned to OAM traffic. Networks that use some form of

L2 or L3 service virtualization would also use exactly the same virtualization for OAM traffic - consider OAM traffic to be just like yet another subscriber class. Example are L2-VPLS or L3-MPLS/VPN instances for OAM traffic. When diverse parts of a network use different - or no - service isolation, the setup of a virtualized OAM plane often becomes very complex: Virtualizing the OAM plane in MPLS/VPN P router (that are not meant or not capable to act as PE routers) need to use a more lightweight OAM virtualization than L3 VPN networks. When the OAM plane needs to be extended beyond the PE side into the access space (e.g. enterprise services or broadband aggregation equipment), MPLS/VPN virtualization is also often undesired. When equipment of different subscriber classes needs to be managed, the OAM VPN has to use complex configurations such as VPN extranet, which itself introduces security challenges.

With the complexity of these approaches, they are in generally only used to protect against operations known upfront to happen predictably. Unexpected OAM operations are in result then not protected. For example, if OAM and subscriber transport routing isolation is not seen as important, changes in subscriber routing configuration becomes difficult and has to be extremely well planned upfront to not impact OAM reachability. Whether this is restructuring of IGP areas, changed from one IGP to another (as happening in enterprises), or more likely a change in routing security or other policy configurations. Any mistake in the process may impact OAM connectivity and require to fallback to an earlier checkpoint, potentially with long interruptions due to equipment reboots.

When not building a completely separate DC today, every network operator tries to come up with a "sufficient" compromise between the different options available: 1) Extra equipment ("DCN" like) to provide additional network connectivity only required for OAM, 2) In-band OAM traffic with security isolation towards subscriber traffic and 3) further isolation/virtualization between OAM and subscriber traffic.

Examples of more lightweight DCN connectivity are emergency network connectivity into SP PoP (Points of Presence) locations and additional "jumpstart" servers holding device firmware/software used to connect to all network equipment, especially to perform initial install and later software/system-firmware upgrade operations. Even though all equipment has TCP/IP OAM connectivity, its ability to support remote initial (greenfield) deployment varies widely between vendors and even different product types of the same vendor. Often, still serial console access is required today or if it can be done via TCP/IP. A local jumpstart server is required because protocols like TFTP are used that do not use windowing.

1.1.5. Zero Touch Device Deployment

Zero-touch deployment Software Solutions have been developed by various vendors, but not only are they most often vendor proprietary, they also depend in most cases on the ability to connect the new equipment to a LAN on which the equipment would receive an IP address via DHCP, as well as additional DHCP or DNS information pointing it to a bootstrap server to connect. Traditionally, DHCP and DNS are set up only to service subscriber devices, so the addresses handed out to them are often not the addresses desired to be used by actual infrastructure equipment. On transit links between network equipment, addressing most often relies on explicit management of addresses instead of DHCP. In addition, to make this approach work, every router has to be set up to proxy DHCP (and potentially DNS) requests and therefore also already have enough network connectivity (including routing) set up to support this. When used in for example broadband aggregation ring topologies, this causes a strict

serialization of deployment hop-by-hop: Every node that needs to act as a proxy for the next node first has to be provisioned with all the services to be able to act as such a proxy.

The complexities and mutual dependencies introduced by all these different aspects of OAM connectivity for ongoing operations as well as initial device bring up are a relevant part of ossification of transport network infrastructures. They are slowing down network operator's ability to evolve networks incrementally to adopt to market and subscriber requirements. They also make security difficult to maintain.

1.1.6. The Impact Of SDN

SDN (Software Developed Network) approaches aim to improve network ability by providing more automation of network provisioning and operations. When SDN applications are written without fully understanding all the above intricacies, they will not be able to safely do their job. Instead, they can disconnect themselves from the equipment they want to manage (or devices behind that device), break security or subscriber services. A simple example are ACL objects with policy lines for both OAM and subscriber traffic. Every simple subscriber service SDN application needs to be written to understand/ignore and not break such OAM policy lines. SDN application that start to model all the aspects of a network including how its OAM is set up may be able to abstract these problems out of higher layers of the SDN application, but as long as the OAM plane setup is not further standardized, such SDN infrastructures will be very network specific and therefore likely not available unless not self-built.

1.2. Intelligent, self-configuring network services

Distributed routing protocols are one of the key ingredients for the success of TCP/IP networks. They provided resilience and flexibility not known before in networking. Yet in TCP/IP. complete network wide auto-addressing did not materialize, so addressing became a first-level requiring manual management. To this day this stays an argument why L2 networks are easier to operate than L3 networks. Only recently limited scope networks such as IETF homenet are trying to solve this in a standardized way.

When more and more network services were introduced over the last decades into TCP/IP networks, each of them had to resort to reinventing the same common underlying infrastructure elements. Because these elements are easier to implement when they are not self-configuring, but instead rely on many operator-provided magic configuration parameters, this has led to more unnecessarily complex and error-prone methods of configuration and often more insecure.

How do services discover their peers on direct or indirect neighbors today? Answer: protocols from L2 (bridging), L3 unicast routing, multicast routing, encryption services, VPN services and OAM services discover direct neighbors with their own neighbor discovery protocols, and as soon as there is no direct L2 adjacency, additional configuration is required. This is clearly not a good, but only a historically evolved state of the art.

1.3. Security in network services

How do protocols secure their communication against infrastructure attacks? Every protocol implements its own encryption/security layer, most often only supporting shared secrets and often required to be explicitly configured on every device.

The per-service complexity resulting from this history is arguably one relevant aspect causing delay in adoption of network services. They are complex and fragile because of many "nerd knobs" (configuration parameters) – especially when security needs to be added to them.

Most often, security has also been added to network services as an afterthought. Important protocols such as SNMP, TFTP, DHCP, DNS, Radius or even HTTP (as used in OAM operators) all have their security challenges. Even when there are widely support encryption options (e.g.: HTTPs), management of keying material stays a challenge.

The state of security for the network infrastructure and its services is especially a problem. Today, every "script-kiddy" can perform attacks by downloading fully automated tools from the Internet to perform attacks. Operators, on the other hand, are left to an overwhelming set of provisioning problems of complex elements to protect their infrastructure.

2. Autonomic Networking: The Vision

Reviewing the problems described above, the vision of autonomic networking was created:

> Build an-in band virtualized DCN that does not suffer from the above described ad-hoc design problems. Make the virtual DCN work the same way hop-by-hop, whatever the role of a device is - whether it is a P, PE or aggregation space device, operating at L3 and/or L2.

The virtual DCN must be completely auto-self-configuring and any configuration it creates on the network devices needs to be independent of and non-changeable by any operator or SDN application actions: "Indestructible" by Operator/SDN actions.

We ended up calling this in-band virtual DCN the "Autonomic Control Plane" (ACP) and call the rest of the routers services/configuration the "Data Plane" (DP) to distinguish it from the ACP. The overall infrastructure to support both DCN like connectivity as well as supporting better intelligent autonomic services is called the “Autonomic Networking Infrastructure.”

To be indestructible, the ACP needs to have auto-configured addressing independent of Data Plane addressing. The ACP also needs to be self-protecting by being encrypted hop-by-hop so that any non-self-secured protocols such as SNMP, TFTP, DHCP, DNS, and Radius would be protected by it. Any new network services would not need to re-invent their own protocol security, because they could communicate with their protocol peers via the ACP, inheriting ACP security.

Finally, the ACP should allow secure zero-touch bootstrap of devices without requiring any Data Plane configuration.

2.1. Example Workflow benefits

What could you do with such an autonomic network?

2.1.1. Brownfield Network

You have an existing network. On every device, you just enable the ACP through a simple command without parameters and the ACP would run - without any other changes to your network. You connect the ACP in the NOC to your back-end equipment and that equipment can now use the ACP to perform OAM operations. Consider a simple manual remote management operation via SSH. You reconfigure some ACL, routing, policies or other aspects of the router. Maybe you erase all running configuration. The device would stop to perform for the data plane and user traffic would stop going through it. Devices in the topology behind this "broken" device may become unreachable in the DP. Using the ACP, you can still reach this device and any device behind it.

2.1.2. Greenfield deployment

Consider rolling out a new aggregation network topology with multiple devices in a ring. Assume the devices are all completely unconfigured, delivered by the manufacturer to locations without any pre-staging. The devices simply need to be wired up with each other, the ACP builds automatically and the devices can be configured from the NOC independently of each other: Reachability of a device further away in the topology can be managed via the ACP without any prior DP configuration in the devices leading to it.

More examples will become clear further down in the document when we explain further details.

3. How does it work (Functional Overview)

3.1. Autonomic Control Plane

The ACP itself is most easily understood as yet another "virtual routing and forwarding" (VRF) context in each device except that it does not provide any configuration but is self configuring.

3.1.1. IPv6 Only

For many reasons in the design and for overall simplicity, only IPv6 is supported. Every router has only one auto-configured address in the ACP on a specific loopback interface. On every interface the ACP probes for ACP capable neighbors and builds a secured virtual p2p tunnel to that neighbor. On the tunnel interfaces, only link-local scope IPv6 addresses are used.

3.1.2. RPL Routing Protocol

The ACP is running a routing protocol called "Routing Protocol for Low Power and Lossy Networks" (RPL). It was originally designed for IoT environments, where it needs to be able to run on constrained devices. RPL has only been defined for IPv6. RPL uses enhanced distance vector (DV) routing calculation which makes it ideal for auto-configuration, because it can scale to a very large number of devices in arbitrary topologies without the need of administrative subdivision of large networks into "areas" (as is common in SPF protocols like IS-IS or OSPF).

RPL can be set up in many ways. For the ACP, it is auto-configured to be as lightweight as possible. It will create a routing table like the Spanning Tree Protocol (STP): a simple tree rooted in the NOC. RPL further minimizes this routing table. Instead of having a separate /128 route to the loopback of every other ACP device, it sets up a default route on the interface towards the root/NOC and only signals to its

neighbors and keeps in its routing table the /128 routes towards ACP devices not reachable via the interface towards the root. The further towards the edge a device is, the smaller its routing table gets.

The ACP routing is therefore automatically highly scalable and lightweight, but does not provide shortest paths between any two ACP devices, but it does provide shortest paths between the root/NOC and any device.

3.1.3. Customer Domain Certificate based Security

Security of the ACP simply relies on public key certificates and implements an automated version of standard public key certificate enrollment procedures. The seed router of an autonomic network acts as a so-called autonomic registrar, which is an enhanced version of a public key enrollment Registration Authority (RA). It gets preconfigured with a target domain name of the ACP and a trust relationship to a Certificate Authority (CA) from which to get certificates. In its most simple instance, the registrar has a built-in CA functionality further simplifying the seed configuration for an autonomic network.

3.1.4. Zero Touch Secure Enrollment

When a device boots Greenfield or when it is already configured, a simple "autonomic" configuration command is issued and the device will first try to acquire an autonomic network certificate. This is called enrollment. It runs a discovery protocol to its neighboring routers. If any of these routers is already part of the autonomic network, it will act as a proxy carrying the certificate enrollment request across the ACP to the registrar. Through the registrar, the enrollment request is authenticated, ideally because the enrolling device has a secure IDevID (Initial Device Identity) such as a Manufacturer Certificate. The authentication of devices can then securely be derived from the list of ordered device serial numbers from the device vendor order confirmation.

3.1.5. Automatic Addressing

The certificates will carry as fields in the subject name the name of the autonomic domain, a unique identity of the registrar, for example, the registrar's primary MAC address and a serial number from the registrar. This scheme allows it to operate multiple registrars in an autonomic network. Each autonomic device automatically assigns its address in the ACP by forming a so-called ULA address from the domain-name, the registrar ID and serial number. "Unique Local Addresses" (ULA) are the IPv6 equivalent of IPv4 RFC1918 local addresses but with the benefit that there are so many of them that by encoding the domain name into the ULA-prefix any two independently chosen domain names will result in non-overlapping ACP addresses.

3.1.6. Automatic hop-by-hop secure ACP channels

When ACP devices auto-discover neighbors that have an autonomic certificate with the same domain, they build a tunnel secured by a negotiated encryption mechanism supported by both devices and mutually authenticated via the devices ACP domain certificate. In its most simple form, this will be IPSec as it is the designated IETF protocol for network layer security. We also expect that negotiation of 802.1ae/MACsec will become a more interesting option in the future when more and more devices will have HW-support for this encryption in their Ethernet chips.

3.1.7. Service Discovery

The ACP also needs to support service discovery through an appropriate ACP-wide protocol. For the AC itself to work, registrars need to announce themselves as registrars so that the enrollment proxy function in every router can do its job and help newly enrolling devices to enroll. Beyond this self-use, service discovery can help either the auto-configuration of services in the NOC to use the ACP and/or intelligent self-configuring services on ACP enabled devices to discover each other and auto-configure their services.

3.2. ACP and NOC/SDN services integration

Most NOC services today require some configuration of these servers on every device in the network. By using service discovery across the ACP, this can be automated and made more resilient, especially when servers in the NOC fail and for some reason not be recreated on the same IP addresses hardcoded in every device in the network.

The simplest service to consider is announcement of NOC authentication services using Radius, Diameter and TACACS. Especially and authentication service that would be used to authenticate management connections (SSH, netconf, HTTPs) into greenfield devices: Once an otherwise unconfigured network device enrolls into the autonomic network and builds the ACP, it would need to be configured from the NOC via such a management configuration connection, and that connection even though it is running across the secure ACP must be authenticated.

Other services of interest include auto-configuration of clock/timing - NTP on every router, syslog, SNMP or JSON/XML/REST based reporting to the NOC or even legacy mechanisms such as TFTP servers to automatically download configuration in newly booted devices.

Once such basic service integration exists, it is easy to use the ACP to create automated workflows with management stations such as SDN controllers. The SDN controller can retrieve reporting messages (syslog, SNMP, REST) when the ACP has discovered a new device and wants to enroll it. The SDN controller can become the policy definition point to permit whether a device is to be permitted into the ACP, and once a device is enrolled, another reporting message can tell the SDN controller that the device is now reachable under its ACP address. The SDN controller can then manage through SSH/netconf or other means dynamically the provisioning and service management of the device via the ACP. Once the data-plane is configured by the SDN controller can also use it as well.

3.3. Distributed autonomous services.

As mentioned before, autonomous services should enable easier creation of autonomous services in the network. We give two examples.

3.3.1. Auto-Security for network protocols

As explained above, the ACP automatically encrypts its hop-by-hop connections. This is made very simple because of the fact that all network devices within the autonomic domain have public key certificates specifically for this domain they can mutually trust and use as authenticators for the security associations. In the same way that the ACP can be secured, autonomic security can easily be extended into protecting all type of protocols running in the data-plane: IGPs, BGP or hop-by-hop link-layer encryption via 802.1ae/MACsec, to name a few. Per-hop link encryption can make higher layer protocol

e.g. routing protocol) encryption redundant and is especially useful when devices are to be deployed in non-secure locations, e.g. co-located in exchanges, on customer premises, or outside (e.g. in last mile aggregation).

3.3.2. Autonomic Fault Management

Fault Management is an ongoing challenge for large network operations. Every network device may require a significant amount of ongoing monitoring of operational parameters from CPU, memory, per-process utilization, diagnostics of all type of network interface health parameters, as well as passive tracking and active probing of hardware components to determine their health. Trying to define these work-flows by exposing all type of parameters and possible active probes to NOC located central intelligence makes it difficult to achieve agility in development, as well as scale and performance in deployment. Instead, low level functions of active HW probing, parameter polling with aggregation and thresholding based reporting of only relevant events is best done in intelligent agents running inside the target devices.

The ability to dynamically update these agents independent of the operating system is important for agile development and deployment of such agents. Decentralized devices for report aggregation running more intelligent agents can correlate events from multiple adjacent devices. Ultimately, autonomic fault diagnostics should be able to analyze failure situations so that the failed SW and/or HW unit can be identified and service personnel can be dispatched to perform a repair (if hardware) or the NOC gets alerted with as much as possible pre-pinpointed error root cause.

The ACP can serve as the reliable and resilient communication fabric for this type of fault management, allowing for fault management communications to not depend on operator/SDN configured data-plane (which by itself can be at fault). The ACP already secures all its communications. A full autonomic network would provide common infrastructure for autonomic agents such as those that an autonomic fault management could be built from.

https://www.ietf.org/archive/id/draft-mtoy-anima-self-faultmang-framework-00.txt provides an overview of a possible framework for such an autonomic fault management.

4. Standardization

The IETF "Autonomic Networking Integrated Model and Approach" working group (ANIMA-WG) is standardizing Autonomic Networking. Its initial charter goal is to standardize the components of what is called the Autonomic Networking Infrastructure. It consists of the ACP, the protocols and functions to perform zero-touch secure enrollment of devices (as explained above), and finally a common signaling protocol (GRASP) that should be able to support common signaling requirements in the ACP, Service Discovery as mentioned above and any agent-to-agent or agent-to-NOC signaling mechanisms. Once those charter goals have made enough progress other aspects of autonomous networking will hopefully be addressed such as Autonomic Agents and network-wide, intent based provisioning.

5. Comparison with other approaches

Self-configuration has become quite important in newer type of networks that are not based on the premise that they can afford or need professional management.

5.1. IETF Homenet

The IETF Homenet working group has developed an IPv6 only self-configuring network solution. The big difference over autonomic networking is that a Homenet does not need to scale to any similar size, and that it does not introduce an automatically built virtual OAM network, but that it simply automates the complete Data Plane. Therefore the functionality of the Data Plane if of course very simple compared to professionally managed networks. It is effectively as if you had only the ACP but no Data-Plane anymore. Homenet is also an L3-device only design. Pure L2 switches would not be an integral part of a homenet (but simply be transparent to it.) In Autonomic Networking, the L2 switch would act as an ACP router to make it manageable via the ACP, but it will continue to operate as an L2 switch in the DP.

5.2. IoT Networks

IoT networks do most often embody the same routing design principles as the ACP when they rely on IPv6 and especially when they use RPL. Like Homenet, those IoT networks use this approach solely to build their data plane. There is no separate OAM plane. The addressing schemes in IoT networks vary. Sometimes they do also use the unique-per-device loopback address routing as the ACP with different way to determine those addresses. Most notably, IoT networks still do not embody strict security models with certificates enrolled in the customer network as the ACP does. Either hard-coded, pre-shared secrets are used, or manual labor involving pairing (as in Bluetooth) or at best solely vendor certificates, therefore making it difficult to recognize rogue devices (acquired by an intruder from the same vendor.)

5.3. Web/Cloud managed network equipment

Web/Cloud provisioned/configured network products are becoming more popular in commercial and small-enterprise offerings, which sometimes are also used by SPs for managed service offerings. These models are all built around the necessity that the cloud service can always reach and manage the device. Because these offerings do not use a clear OAM (ACP)/DP separation, it is crucial that the web/cloud part fully controls all possible configurations such that the network connectivity to the cloud service is not interrupted. These types of services are still evolving, so the complexity of service configuration options and topologies supported are still constrained enough that this approach works very well. Especially when these services expand into more complex customer side topologies, it will become more complex such that any impact the configuration of one device has on the reachability of other devices is taken into account. This can lead to complexity of the cloud service that is likely more easily resolved by introducing the OAM/DP separation that the ACP offers.

6. ACP implementation design

6.1. Classic Router-OS ACP integration

To make the ACP as indestructible and independent of the devices actual software as possible, the design of its implementation has a big impact. In non-modular device software environments, it can be challenging to separate out the ACP fully. Instead the ACP can be implemented inside the actual operating system software, for example in a router software environment as one of the already supported VRFs except that its configuration is hidden, indestructible and automatically created. This configuration includes all those features explained above, some of which may require new development work (RPL routing, IPv6, loopback addresses, service discovery, automatic building of tunnels, certificate enrollment/registrar functions).

While this approach provides the easiest way to implement the ACP on these (legacy OS) type of devices, it makes it more difficult to make the ACP indestructible against misbehavior in the router software and may impact the ability to stay indestructible under all type of low-level configuration commands. Even more so, this approach does not permit to build the ACP before the actual router software is fully booted and therefore allowing using the ACP to be used for much lower level device management. In operations that otherwise require direct console serial console connection such as uploading new operating system software, especially when there is no working system software on local storage, as well as remotely controlling/observing the actual software boot process and controlling it (e.g. during upgrades.)

6.1. Hypervisor Integration

In more modern device software architectures where some "lightweight" host/hypervisor OS is used and the router software is running as VMs or containers, the ACP is best implemented inside that hypervisor/lightweight OS. If the OS/hypervisor is derived from Linux, the ACP can be implemented as a set of Linux applications completely independent of any other router software and then become a "Front-End Network" solution, available even for remote inland bootstrap device management.

6.2. BMC Integration

On PC/Data-Center servers, it is common to have additional hardware for out-of-band management of the PC. Some high-end network devices have similar "front-end-management-CPU" designs. This so-called "Baseboard Management Controller" BMC provides virtual connections into the device as low level as BIOS to provide device level management. IPMI is one industry standard protocol for BMC. When network device hardware can afford to have similar BMC hardware, one can consider implementing the ACP inside such a BMC.

The key difference the ACP targets to achieve compared to standard PC-server BMC level management is the automatic in-band network connectivity. BMCs at best can share one or two baseboard Ethernet interfaces with the main system and this is done through dedicated hardware. More often than not, BMCs are connected via separate Ethernet ports.

In network devices, the ACP would need to be able to access all network interfaces to build it’s in-band connectivity. In most network devices, those network interfaces are serviced through accelerated hardware functions: FPGA, ASIC, forwarding plane microcode CPU. Enabling ACP channels to be defined through that additional hardware/forwarding-plane functions is the main challenge when the ACP implementation does not leverage the existing router-OS, but is implemented standalone - to be indestructible by the router-OS operations.

6.3. Network Platform Evolution

The trend for all but the highest end of network devices to turn from hardware acceleration to high-performance x86 software forwarding (with projects like fd.io and others) makes solutions easier. Separation of ACP channels could for example be achieved very low level in the hardware by using virtual PCI interfaces supported today on many Ethernet MIC chips in data center compute hardware. This makes it possible that the main router software both see separate, non-conflicting PCI Ethernet interfaces - each one with its own separate MAC address.

7. SNBI: An Open Source Implementation of Autonomic Networking Infrastructure

Beside commercial implementations, we have also worked on an evolving open source reference implementation of bootstrap and ACP. This work is done as a project in Open Daylight, called the "Secure Network Bootstrapping Infrastructure" (SNBI): https://wiki.opendaylight.org/view/SecureNetworkBootstrapping:Main

The current (Q2/2016) implementation provides a Java/Karaf based autonomic registrar and a Linux based implementation of the ACP bundled together as a docker container. It could be used as the basis for low-level ACP implementation in network device platforms as described above.

Conclusion

This paper gave an overview of the Autonomic Networking Infrastructure with the Autonomic Control Plane at its core. ANI/ACP are a solution to provide a unified, secure, auto-configuring OAM infrastructure across any type of network devices and topologies. ANI/ACP can provide a replacement for traditional DCN, and avoid developing and managing ad-hoc OAM plane isolation and security in TCP/IP network such as MSO core, distribution and access networks.

An automatically built, reliable/secure and indestructible OAM plane is crucial to overcome stagnation of network infrastructures. It enables both the easier development of more intelligent in-network services such as improved security or fault management, and it enables more SDN-centric provisioning of network configuration to avoid harming itself by breaking the network connectivity it relies itself on to manage network elements.

ANI exists in commercial implementations, e.g.: Cisco. Initial Open Source reference implementations exist as well. Based on the implementation approach of ACP/ANI it could also evolve to become a front-end-network component of BMC-style device level management, allowing remote HW/SW maintenance even when no working router-SW is available on a platform.

Abbreviations

ACP	Autonomic Control Plane
ANI	Autonomic Networking Infrastructure
DCN	Data Communication Network – a separate OAM network used in large Telco's.
PAD	Packet Assembler/Disassembler (X.25 Terminal Server)
RPL	Routing Protocol for Low power and Lossy Networks

Bibliography & References

https://trac.tools.ietf.org/wg/anima/trac/wiki - IETF ANIMA (Autonomic Networking Integrated Model and Approach) working group WiKi.

https://datatracker.ietf.org/wg/anima/documents/ - Current ANIMA working group drafts

https://www.ietf.org/archive/id/draft-mtoy-anima-self-faultmang-framework-00.txt

https://wiki.opendaylight.org/view/SecureNetworkBootstrapping:Main - Homepage of Open Source Autonomic Networking implementation (incomplete).

https://www.ciscolive.com/online/connect/search.ww - BRKGEN-2999, BRKSDN-2047. Presentations giving overview and deployment guidance to Cisco's commercial implementation of autonomic networking.

