Internet Research Task Force
Internet-Draft
Intended status: Informational
Expires: June 23, 2014

M. Behringer
M. Pritikin
S. Bjarnason
A. Clemm
Cisco Systems
B. Carpenter
Univ. of Auckland
S. Jiang
Huawei Technologies Co., Ltd
L. Ciavaglia
Alcatel-Lucent
December 20, 2013

Autonomic Networking - Definitions and Design Goals
draft-irtf-nmrg-autonomic-network-definitions-00.txt

Abstract

Autonomic systems were first described in 2001. The fundamental goal is self-management, including self-configuration, self-optimization, self-healing and self-protection.

This document applies the concepts of autonomic systems to a network, and describes the definitions and design goals of Autonomic Networking. The goal is a network where nodes have minimal dependencies on human administrators or centralized management systems.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on June 23, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction to Autonomic Networking 2
2. Definitions 3
3. Design Goals 4
3.1. Self-Management 4
3.2. By Default Secure 5
3.3. Decentralisation and Distribution 5
3.4. Simplification of the Northbound Interfaces 5
3.5. Abstraction 6
3.6. Autonomic Reporting 6
3.7. Modularity 6
3.8. Independence of Function and Layer 7
3.9. Full Life Cycle Support 7
4. Non Design Goals 7
4.1. Eliminate human operators 8
4.2. Eliminate emergency fixes 8
4.3. Eliminate management control and central policy 8
4.4. Eliminate existing configuration tools 8
4.5. Eliminate existing network management systems 8
5. Guidelines for Case Studies 9
6. An Autonomic Reference Model 9
7. Security Considerations 10
8. Acknowledgements 10
9. Informative References 10
Authors' Addresses 11

1. Introduction to Autonomic Networking

Autonomic systems were first described in a manifesto by IBM in 2001 [Kephart]. The fundamental concept involves eliminating external systems from a system's control loops and closing of control loops within the autonomic system itself, with the goal of providing the

autonomic system with self-management capabilities, including self-configuration, self-optimization, self-healing and self-protection.

IP networking was initially designed with similar properties in mind. An IP network should be distributed and redundant to withstand outages in any part of the network. A routing protocol such as OSPF or ISIS exhibits properties of self-management, and can thus be considered autonomic in the definition of this document.

However, as IP networking evolved, the ever increasing intelligence of network element was often not put into protocols to follow this paradigm, but into configuration. This configuration made network elements highly dependent on some process that manages them, either a human, or a network management system.

Autonomic Networking aims at putting the intelligence of today's operations back into algorithms at the node level, to minimize dependency on human administrators and central management systems. Some information an autonomic node requires however cannot be discovered; where input from some central intelligence is required, it is provided in a highly abstract, network wide form.

This document provides the definitions and gesign goals for Autonomic Networking.

2. Definitions

Autonomic: Self-managing (self-configuring, self-protecting, self-healing and self-optimizing); however, allowing high-level guidance by a central entity, through intent.

Intent: An abstract, high level policy used to operate the network autonomically. Its scope is an autonomic domain, such as an enterprise network. It does not contain configuration or information for a specific node. It may contain information pertaining to nodes with a specific role.

Autonomic Domain: A collection of autonomic nodes that instantiate the same intent.

Autonomic Function: A function which requires no configuration, and can derive all required information either through self-knowledge, discovery or through intent.

Autonomic Service Agent: An agent implemented on an autonomic node which implements an autonomic function, either in part (in the case of a distributed function) or whole.

Autonomic Node: A node which employs autonomic functions. It may operate on any layer of the networking stack. Examples are routers, switches, personal computers, call managers, etc.

Fully Autonomic Node: A node which employs exclusively autonomic functions. It requires no configuration.

Autonomic Network: A network containing autonomic nodes.

Fully Autonomic Network: A network consisting of exclusively fully autonomic nodes.

3. Design Goals

This section explains the high level goals of Autonomic Networking, independent of any specific solutions.

3.1. Self-Management

The original design goals of autonomic systems as described in [Kephart] also apply to Autonomic Networks. The over-arching goal is self-management, which is comprised of several self-* properties. The most commonly cited are:

- o Self-configuration: The nodes do not require to be configured, but they configure themselves, based on self-knowledge, discovery, and intent. Discovery is the default way for a node to receive the information it needs to operate.

- o Self-healing: The nodes adapt on their own to changes in the environment, and heal problems automatically.

- o Self-optimising: The nodes automatically determine ways to optimise their behaviour.

- o Self-protection: The nodes automatically secure themselves against potential attacks.

Almost any network can be described as "self-managing", as long as the definition of "self" is large enough. For example, to a residential user, the service provider network she connects to could be considered "autonomic", because the user only specifies a very high level policy such as "Internet access" and is not exposed to any internals of the network.

For the work in the IETF and IRTF we define the "self" properties on the node level. It is the design goal to make network nodes self-managing, in other words, minimally dependent on management systems

or controllers, as well as human operators. Self-managing nodes might need to exchange information with other nodes in order to achieve the required goals.

3.2. By Default Secure

All autonomic interactions should be by default secure. This requires that any member of an autonomic domain can assert its membership using a domain identity, for example a certificate issued by a domain certification authority. This domain identity is used for nodes to learn about their neighbouring nodes, to determine the boundaries of the domain, and to cryptographically secure interactions within the domain. Nodes from different domains can also mutually verify their identity and secure interactions as long as they have a common trust anchor.

A strong, cryptographically verifiable domain identity is a fundamental cornerstone in autonomic networking. It can be leveraged to secure all communications, and allows thus automatic security without traditional configuration, for example pre-shared keys.

Autonomic nodes must be able to adapt their behaviour depending on the domain of the node they are interacting with.

3.3. Decentralisation and Distribution

The goal of Autonomic Networking is to minimise dependencies on central elements; therefore, de-centralisation and distribution are fundamental to the concept. If a problem can be solved in a distributed manner, it should not be centralised.

In certain cases it is today operationally preferable to keep a central repository of information, for example a user database on a AAA server. An autonomic network must also be able to use such central systems, in order to be deployable. However, it is possible to distribute such databases as well, and such efforts should be at least considered.

3.4. Simplification of the Northbound Interfaces

Even in a decentralised solution, certain information flows with central entities are required. Examples are the definition of intent or high level service definitions, as well as network status requests and aggregated reporting.

Therefore, also elements in an autonomic network require a northbound interface. However, the design goal is to maintain this interface as simple and high level as possible.

3.5. Abstraction

An administrator or autonomic management system interacts with an autonomic network on a high level of abstraction. Intent is defined at a level of abstraction that is much higher than that of typical configuration parameters, for example, "optimize my network for energy efficiency". Intent must not be used to convey low-level commands or concepts, since those are on a different abstraction level. The administrator should not even be exposed to the version of the IP protocol running in the network.

Also on the reporting and feedback side an autonomic network abstracts information and provides high-level messages such as "the link between node X and Y is down".

3.6. Autonomic Reporting

An autonomic network, while minimizing the need for user intervention, still needs to provide users with visibility like in traditional networks. However, in an autonomic network reporting should happen on a network wide basis. Information about the network should be collected and aggregated by the network itself, presented in consolidated fashion to the administrator.

The layers of abstraction that are provided via intent need to be supported for reporting functions as well, in order to give users an indication about the effectiveness of their intent. For example, in order to assess how effective the network performs with regards to the intent "optimize my network for energy efficiency", the network should provide aggregate information about the number of ports that were able to be shut down while validating current service levels are on aggregate still met.

Autonomic network events should concern the autonomic network as a whole, not individual systems in isolation. For example, the same failure symptom should not be reported from every system that observes it, but only once for the autonomic network as a whole. Ultimately, the autonomic network should support exception based management, in which only events that truly require user attention are actually notified. This requires capabilities that allow systems within the network to compare information and apply special algorithms to determine what should be reported.

3.7. Modularity

It is unrealistic to expect a fully autonomic network in complex environments for many years to come. While simple networks may

become autonomic in one single step, a phased approach is required for most of today's networks.

Autonomic functions can be implemented in a modular way. For example, the internal routing algorithm in many networks today is already mostly autonomic. Other modules can be made autonomic step by step.

3.8. Independence of Function and Layer

Today's autonomic functions may reside on any layer in the networking stack. For example, layer 2 switching today is already relatively autonomic in many environments; routing functions can be autonomic. "Autonomic" in the context of this framework is a property of a node. This node can be a switch, router, server, or call manager. Autonomic functionality is independent of the function of a node. Even application layer functionality such as unified communications can be autonomic.

An Autonomic Network requires an overall control plane for autonomic nodes to communicate. As in general IP networking, IP is the layer that binds all those elements together; autonomic functions in the context of this framework should therefore operate at the IP layer. This concerns neighbour discovery protocols and other autonomic control plane functions.

3.9. Full Life Cycle Support

An autonomic node does not depend on external input to operate; it needs to understand its current situation and surrounding, and operate according to its current state. Therefore, an autonomic node must understand its full life cycle, from first manufacturing testing through deployment, testing, troubleshooting, up to decommissioning.

The state of the life-cycle of an autonomic node is reflected in a state model. The behaviour of an autonomic node may be different for different deployment states.

4. Non Design Goals

This section identifies various items which are explicitly not design goals for autonomic networks, which are mentioned to avoid misunderstandings of the general intention.

4.1. Eliminate human operators

The problem targeted by autonomic networking is the error-prone and hard to scale model of individual configuration of network elements, traditionally by manual commands but today mainly by scripting and/or configuration management databases. This does not, however, imply the elimination of skilled human operators, who will still be needed for oversight, policy management, diagnosis, reaction to help desk tickets, etc. etc. The main impact on operators should be less tedious detailed work and more high-level work. (They should become more like doctors and nurses than hospital orderlies.)

4.2. Eliminate emergency fixes

However good the autonomous mechanisms, sometimes there will be fault conditions etc. that they cannot deal with correctly. At this point skilled operator interventions will be needed to correct or work around the problem. Hopefully this can be done by high-level mechanisms (adapting the policy database in some way) but in some cases direct intervention at device level may be unavoidable. This is obviously the case for hardware failures, even if the autonomic network has bypassed the fault for the time being. Truck rolls will not be eliminated when faulty equipment needs to be replaced. However, this may be less urgent if the autonomic system automatically reconfigures to minimise the operational impact.

4.3. Eliminate management control and central policy

Senior management might fear loss of control of an autonomic network. In fact this is no more likely than with a traditional network; the emphasis on automatically applying general policy and security rules might even provide more management control.

4.4. Eliminate existing configuration tools

While autonomic networks will rarely need manual intervention, there is no expectation that traditional top-down configuration tools will vanish immediately. Autonomic techniques will have to co-exist with them, and they will survive for as long as they are useful. Initially they will certainly play a part in confidence-building in the autonomic method, and they will be held in reserve for emergency use for a long time.

4.5. Eliminate existing network management systems

Existing monitoring and reporting systems will continue to be needed, and as just noted existing configuration mechanisms will not vanish. Therefore, it is to be expected that the existing NMS will be

retained in parallel with autonomic mechanisms, and will be adapted as necessary. Some aspects of the autonomic mechanism (e.g. aggregated reporting, exception reporting) should indeed be integrated with the existing NMS as far as possible.

5. Guidelines for Case Studies

[This section is work in progress.]

6. An Autonomic Reference Model

An Autonomic Network consists of Autonomic Nodes. Those nodes communicate with each other through an Autonomic Control Plane which provides a robust and secure communications overlay. The Autonomic Control Plane is self-organizing and autonomic itself.

An Autonomic Node contains various elements, such as autonomic service agents. Figure 1 shows a reference model of an autonomic node. The elements and their interaction are:

- o Autonomic Service Agents, which implement the autonomic behaviour of a specific service or function.

- o Self-knowledge: An autonomic node knows its own properties and capabilities

- o Network Knowledge (Discovery): An autonomic service agent may require various discovery functions in the network, such as service discovery.

- o Intent: Network wide high level policy. Autonomic Service Agents use an intent interpretation engine to locally instantiate the global intent. This may involve coordination with other Autonomic Nodes.

- o Feedback Loops: Control elements outside the node may interact with autonomic nodes through feedback loops.

- o An Autonomic User Agent, providing a front-end to external users (administrators and management applications) through which they can communicate intent, receive reports, and monitor the Autonomic Network.

- o Autonomic Control Plane: Allows the node to communicate with other autonomic nodes. Autonomic functions such as intent distribution, feedback loops, discovery mechanisms, etc, use the autonomic control plane.

```
+-------------------------------------------------------------+
|               +----------+ +--------------+                |
|               |          | | Feedback     |                |
|               | Intent   | |    Loops     |                |
|               +----------+ +--------------+                |
|                         ^     ^                             |
|                    Autonomic User Agent                     |
|                         V     V                             |
| +-----------+        +------------+        +------------+  |
| | Self-     |        | Autonomic  |        | Network    |  |
| | knowledge |<------>| Service    |<------>| Knowledge  |  |
| |           |        | Agents     |        | (Discovery)|  |
| +-----------+        +------------+        +------------+  |
|                            ^                    ^          |
|                            |                    |          |
|                            V                    V          |
|-------------------------------------------------------------|
|                 Autonomic Control Plane                     |
|-------------------------------------------------------------|
|           Standard Operating System Functions               |
+-------------------------------------------------------------+
```

Figure 1

7. Security Considerations

This document specifies a framework. Security is an integral part of this framework.

8. Acknowledgements

The work on Autonomic Networking is the result of a large team project at Cisco Systems. In alphabetical order: Ignas Bagdonas, Parag Bhide, Balaji BL, Toerless Eckert, Yves Hertoghs, Bruno Klauser.

The ETSI working group AFI (http://portal.etsi.org/afi) defines a similar framework for autonomic networking in the "General Autonomic Network Architecture" [GANA]. Many concepts explained in this document can be mapped to the GANA framework. The mapping is outside the scope of this document. Special thanks to Ranganai Chaparadza for his comments and help on this document.

9. Informative References

[GANA] ETSI GS AFI 002, , "Autonomic network engineering for the self-managing Future Internet (AFI): GANA Architectural Reference Model for Autonomic Networking, Cognitive Networking and Self-Management.", April 2013, <http://www.etsi.org/deliver/etsi_gs/AFI/001_099/002/01.01.01_60/gs_afi002v010101p.pdf>.

[Kephart] Kephart, J. and D. Chess, "The Vision of Autonomic Computing", IEEE Computer vol. 36, no. 1, pp. 41-50, January 2003.

Authors' Addresses

Michael Behringer
Cisco Systems
Building D, 45 Allee des Ormes
Mougins 06250
France

Email: mbehring@cisco.com

Max Pritikin
Cisco Systems

Email: pritikin@cisco.com

Steinthor Bjarnason
Cisco Systems

Email: sbjarnas@cisco.com

Alex Clemm
Cisco Systems

Email: alex@cisco.com

Brian Carpenter
Department of Computer Science
University of Auckland
PB 92019
Auckland 1142
New Zealand

Email: brian.e.carpenter@gmail.com

Sheng Jiang
Huawei Technologies Co., Ltd
Q14, Huawei Campus
No.156 Beiqing Road
Hai-Dian District, Beijing 100095
P.R. China

Email: jiangsheng@huawei.com

Laurent Ciavaglia
Alcatel-Lucent

Email: Laurent.Ciavaglia@alcatel-lucent.com

Network Management Research Group
Internet-Draft
Intended status: Informational
Expires: August 18, 2014

M. Behringer
Cisco Systems
B. Carpenter
Univ. of Auckland
S. Jiang
Huawei Technologies Co., Ltd

February 14, 2014

Gap Analysis for Autonomic Networking
draft-jiang-nmrg-an-gap-analysis-00

Abstract

This document summarises a problem statement for an IP-based autonomic network that is mainly based on distributed network devices. The document reviews the history and current status of autonomic aspects of IP networks. It then reviews the current network management style, which is still heavily depending on human administrators. Finally the document describes the general gaps between the ideal autonomic network concept and the current network abilities.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 18, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. Terminology 3
3. Current Status of Autonomic Aspects of IP Networks 3
3.1. IP Address Management and DNS 3
3.2. Routing 4
3.3. Configuration of Default Router 4
3.4. Hostname Lookup 5
3.5. User Authentication and Accounting 5
3.6. Security 5
3.7. Miscellaneous 6
4. Current Non-Autonomic Behaviors 6
4.1. Network Establishment 7
4.2. Network Maintenance & Management 7
4.3. Troubleshooting and Recovery 8
5. Approach toward Autonomy 9
5.1. More Coordination among Devices or Network Partitions . . 9
5.2. Benefit from Knowledge 9
6. Security Considerations 10
7. IANA Considerations 10
8. Acknowledgements 10
9. Informative References 10
Authors' Addresses 12

1. Introduction

The general goals and relevant definitions for autonomic networking are discussed in [I-D.irtf-nmrg-autonomic-network-definitions]. In summary, the fundamental goal of an autonomic network is self-management, including self-configuration, self-optimization, self-healing and self-protection. Whereas interior gateway routing protocols such as OSPF and IS-IS largely exhibit these properties, most other aspects of networking require top-down configuration often involving human administrators and a considerable degree of centralisation. In essence Autonomous Networking is putting all network configuration onto the same footing as routing, limiting manual or database-driven configuration to an essential minimum. It should be noted that this is highly unlikely to eliminate the need for human administrators, because many of their essential tasks will remain. The idea is to eliminate tedious and error-prone tasks, for

example manual calculations, cross-checking between two different configuration files, or tedious data entry. Higher level operational tasks, and trouble-shooting, will remain to be done in any case.

Note in draft: This is a preliminary version. It certainly lacks information about current status, and it lacks many external references. Especially the final section (Section 5) is very preliminary. Comments and suggestions are very welcome.

2. Terminology

The terminology defined in [I-D.irtf-nmrg-autonomic-network-definitions] is used in this document. Additional terms include:

- o Automatic: A process that occurs without human intervention, with step-by-step execution of rules. However it relies on humans defining the sequence of rules, so is not Autonomic in the full sense. For example, a start-up script is automatic but not autonomic.

3. Current Status of Autonomic Aspects of IP Networks

This section discusses the history and current status of autonomy in various aspects of network configuration, in order to establish a baseline for the gap analysis. In one particular area, routing protocols, autonomic information exchange and decision is a well established mechanism. The question is how to extend autonomy to cover all kinds of network management objectives.

3.1. IP Address Management and DNS

Originally there was no alternative to completely manual and static management of IP addresses. Once a site had received an IPv4 address assignment (usually a Class C /24 or Class B /16, and rarely a Class A /8) it was a matter of paper-and-pencil design of the subnet plan (if relevant) and the addressing plan itself. Subnet prefixes were manually configured into routers, and /32 addresses were assigned administratively to individual host computers, and configured manually by system administrators. Records were typically kept in a plain text file or a simple spreadsheet.

Clearly this method was clumsy and error-prone as soon as a site had more than a few tens of hosts, but it had to be used until DHCP [RFC2131] became a viable solution during the second half of the 1990s. DHCP made it possible to avoid manual configuration of individual hosts (except, in many deployments, for a small number of servers configured with static addresses).

In terms of management, it is difficult to separate IP address management from DNS management. At roughly the same time as DHCP came into widespread use, it became very laborious to manually maintain DNS source files in step with IP address assignments. Because of reverse DNS lookup, it also became necessary to synthesise DNS names even for hosts that only played the role of clients. Therefore, it became necessary to synchronise DHCP server tables with forward and reverse DNS. For this reason, Internet Protocol address management tools emerged. These are, however, a centralised and far from autonomic type of solution.

A related issue is prefix delegation, especially in IPv6 when more than one prefix may be delegated to the same physical subnet. DHCPv6 Prefix Delegation [RFC3633] is a useful solution, but how this topic is to be handled in home networks is still an open question. Still further away is automated assignment and delegation of IPv4 subnet prefixes.

Another complication is the possibility of Dynamic DNS Update [RFC2136]. With appropriate security, this is an autonomic approach, where no human intervention is required to create the DNS records for a host. Also, there are coexistence issues with a traditional DNS setup.

3.2. Routing

Since a very early stage, it has been a goal that Internet routing should be self-healing when there is a failure of some kind in the routing system (i.e. a link or a router goes wrong). Also, the problem of finding optimal routes through a network was identified many years ago as a problem in mathematical graph theory, for which well known algorithms were discovered (the Dijkstra and Bellman-Ford algorithms). Thus routing protocols became largely autonomic in the 1980s, as soon as the network was big enough for manual configuration of routing tables to become difficult.

IGP routers do need some initial configuration data to start up the autonomic routing protocol. Also, BGP-4 routers need static configuration of routing policy data. So far, this policy configuration has not been made autonomic at all.

3.3. Configuration of Default Router

Originally this was a manual operation. Since the deployment of DHCP, this has been automatic as far as most IPv4 end systems are concerned, but the DHCP server must be appropriately configured. In simple environments such as a home network, the DHCP server resides in the same box as the default router, so this configuration is also

automatic. In more complex environments, where an independent DHCP server or a local DHCP relay is used, configuration is more complex and not automatic.

In IPv6 networks, the default router is provided by Router Advertisement messages [RFC4861] from the router itself, and all IPv6 hosts make use of it. The router may also provide more complex Route Information Options. The process is automatic as far as all IPv6 end systems are concerned, and DHCPv6 is not involved. Howwever there are still open issues when more than one prefix is in use on a subnet and more than one first-hop router may be available as a result.

3.4. Hostname Lookup

Originally host names were looked up in a static table, often referred to as /etc/hosts from its traditional file path in Unix systems. When the DNS was deployed during the 1980s, all hosts needed DNS resolver code, and needed to be configured with the IP addresses (not the names) of suitable DNS servers. Like the default router, these were originally manually configured. Today, they are provided automatically via DHCP or DHCPv6 [RFC3315]. For IPv6 end systems, there is also a way for them to be provided automatically via a Router Advertisement option. However, the DHCP or DHCPv6 server, or the IPv6 router, need to be configured with the appropriate DNS server addresses.

3.5. User Authentication and Accounting

Originally, user authentication and accounting are mainly based on the physical connectivities. Network operators charged based on the set up of dedicated physical links with users. Autonomic user authentication are introduced by Point-to-Point Protocol [RFC1661], [RFC1994] and RADIUS protocol [RFC2865], [RFC2866] in early 1990s. As long as a user complete online authentication through RADIUS protocol, the accounting for that user starts on AAA server autonomically. This mechanism enables charging business model based on the usage of users, either traffic based or time based. However, the management for user authentication information remains manual by network administrators.

3.6. Security

Security has many aspects that need configuration and are therefore candidates to become autonomic. On the other hand, it is essential that a network’s central policy should be applied strictly for all security configuration. As a result security has largely been based on centrally imposed configurations.

Many aspects of security depend on policy, for example firewall policies. Policies are by definition human made and will therefore also persist in an autonomic environment. However, policies are becoming more high-level, abstracting for example addressing, and focusing on the user or application. The methods to manage, distribute and apply policy, and to monitor compliance and violations could be autonomic.

Today, many security mechanisms show some autonomic properties. For example user authentication via 802.1x allows automatic mapping of users after authentication into logical contexts (typically VLANs). While today configuration is still very important, the overall mechanism displays signs of self-adaption to changing situations.

BGP Flowspec [RFC5575] allows a partially autonomic threat defense mechanism, where threats are identified, the flow information is automatically distributed, and counter-actions can be applied. Today typically a human operator is still in the loop to check correctness, but over time such mechanisms can become more autonomic.

Negotiation capabilities, present in many security protocols, also display simple autonomic behaviours. In this case a security policy about algorithm strength can be configured into servers but will propagate automatically to clients. A proposal has been made recently for automatic bootstrapping of trust in a network [I-D.behringer-default-secure]. Solutions for opportunistic encryption have been defined [RFC4322], [I-D.farrelll-mpls-opportunistic-encrypt], but these do not adhere to a central policy.

3.7. Miscellaneous

There are innumerable other properties of network devices and end systems that today need to be configured either manually or using a management protocol such as SNMP [RFC1157] or NETCONF [RFC6241]. In a truly autonomic network, all of these would need to either have satisfactory default values or be configured automatically. Some examples are parameters for tunnels of various kinds, flows (in an SDN context), quality of service, service function chaining, energy management, system identification, NTP configuration etc. Even one undefined parameter would be sufficient to prevent fully autonomic operation.

4. Current Non-Autonomic Behaviors

In the current networks, many operations are still heavily depending on human intelligence and decision, or on centralised top-down network management systems. These operations are the targets of

Autonomic Network technologies. The ultimate goal of Autonomic Network is to replace tedious human operations by autonomic functions, so that the networks can independently run without having to ask human support for routine details, while it remains possible to restore human intervention when unavoidable. Of course, there would still be the absolute minimum of human input required, particularly during the network establishment stage, and during difficult trouble-shooting.

This section analyzes the existing human and central dependencies in the current networks.

4.1. Network Establishment

Network establishment requires network operators to analyze the requirements of the new network, design a network archetecture and topology, decide device locations and capacities, set up hardware, design network services, choose and enable required protocols, configure each device and each protocol, set up user authentication and accounting policies and databases, design and deploy security mechanisms, etc.

Overall, these jobs are quite complex work that cannot become fully autonomic in the forseeable future. However, part of these jobs may be able to become autonomic, such as device and protocol configurations and database population. The initial network management policies/behaviors may also be transplanted from other networks and automatically localized.

4.2. Network Maintenance & Management

The network maintenance and management are very different for ISP networks and enterprise networks. ISP networks have to change much more frequently than enterprise networks, given the fact that ISP networks have to serve a large number of customers who have very diversified requirements. The current rigid model is that network administrators design a limited number of services for customers to order. New requirements of network services may not be able to be met quickly by human management. Given a real-time request, the response must be autonomic, in order to be flexible and quickly deployed. However, behind the interface, describing abstracted network information and user authorization management may have to depend on human intelligence from network administrators in the forseeable future. User identification integration/consolidation among networks or network services are another challenge for autonomic network access. Currently, the end users have to manually manage their user accounts and authentication information when they switch among networks or network services.

Classical network maintenance and management mainly manages the configuration of network devices. Tools have developed to enable remote management and make the management easier. However, the decision of each configuration depends either on human intelligence or rigid templates. This is the source of most network configuration errors. It is also the barrier to increase the utility of network resources because the human management cannot respond quickly enough to network events, such as traffic bursts, etc. For example, currently, a light load is normally assumed in network design because there is no mechanism to properly handle a sudden traffic flood. It is actually normal to avoid network crashes caused by traffic overload by wasting a huge amount of resources.

Autonomic decision processes of configuration would enable dynamic management of network resources (by managing resource relevant configuration). Self-adapting network configuration would adjust the network into the best possible situation, which also prevents configuration errors from having lasting impact.

4.3. Troubleshooting and Recovery

The current networks are suffering difficulties in locating the cause of network failures. Although the network devices may issue many warnings during running, most of them are not sufficiently precise to be identified as errors. Some of them are early warnings that would not develop into real errors. Others are in effect random noise. For many scenarios, human experience is vital to identify real issues and locate them. This situation may be improved by associating warnings from multiple network devices together. Also, introducing automated learning techniques (comparing current warnings with historical relationships between warnings and actual faults) could increase the possibility and success rate of autonomic network diagnoses and troubleshooting.

Depending on the network errors, some of them may require human interventions, particularly for hardware failures. Meanwhile, some network management behavior may help to reduce the impact from errors, such as switching traffic flows around. Today this is usually manual. Software failures and configuration errors (including to roll back software versions and to reboot hardware) currently depend on humans. Such problems could be autonomically corrected if there were diagnostics and recovery functions defined in advance for them. This would fulfill the concept of self-healing.

5. Approach toward Autonomy

The task of autonomic networking is to build up individual autonomic decision processes that could properly combine to respond to every type of network event. This section (when complete) will outline what needs to be developed.

5.1. More Coordination among Devices or Network Partitions

Events in networks are normally not independent. They are associated with each other. But most of current response functions are based on independent processes. The network events that may naturally happen distributed should be associated in the autonomic processes.

In order to make right or good decisions autonomically, the network devices need to know more information than just reachability (routing) information from the relevant or neighbor devices. There are dependencies between such information and configurations. Currently, most of these configurations currently require manual coordination by network administrators.

There are therefore increased requirements for horizontal information exchanging in the networks. Particularly, negotiation among network devices are needed for autonomic decision. [I-D.jiang-config-negotiation-ps] analyzes such requirements. Although there are many existing protocols with negotiation ability, each of them are only serve a specifc and narrow purpose. [I-D.jiang-config-negotiation-protocol] is one of the attempts to create a generic negotiation platform, which would support different negotiation objectives.

5.2. Benefit from Knowledge

The more knowledge we have, the more intelligent we are. It is the same for networks and network management. It is when one component in the network lacks knowledge that affects what it should do, and another component has that knowledge, that we usually rely on a human operator or a centralised management tool to convey the knowledge.

Up to now, most available network knowledge is only the current network status, either inside a device or relevant data from other devices.

However, historic knowledge is very helpful to make correct decisions, in particular to reducing network oscillation or to manage network resources over time. Transplantable knowledge from other networks can be helpful to initially set up a new network or new network devices. Knowledge of relationship between network events

and network configuration may help network to decide the best parameters according to real performance feedback.

6. Security Considerations

This document is focussed on what is missing to allow autonomic network configuration, including of course security settings. Therefore, it does not itself create any new security issues. It is worth underlining that autonomic technology must be designed with strong security properties from the start, since a network with vulnerable autonomic functions would be at great risk.

7. IANA Considerations

This memo includes no request to IANA.

8. Acknowledgements

This document was produced using the xml2rfc tool [RFC2629].

9. Informative References

[I-D.behringer-default-secure]
Behringer, M., Pritikin, M., and S. Bjarnason, "Making The Internet Secure By Default", draft-behringer-default-secure-00 (work in progress), January 2014.

[I-D.farrelll-mpls-opportunistic-encrypt]
Farrel, A. and S. Farrell, "Opportunistic Encryption in MPLS Networks", draft-farrelll-mpls-opportunistic-encrypt-02 (work in progress), February 2014.

[I-D.irtf-nmrg-autonomic-network-definitions]
Behringer, M., Pritikin, M., Bjarnason, S., Clemm, A., Carpenter, B., Jiang, S., and L. Ciavaglia, "Autonomic Networking - Definitions and Design Goals", draft-irtf-nmrg-autonomic-network-definitions-00 (work in progress), December 2013.

[I-D.jiang-config-negotiation-protocol]
Jiang, S., Carpenter, B., Liu, B., and Y. Yin, "Configuration Negotiation Protocol for Network Devices", draft-jiang-config-negotiation-protocol-00 (work in progress), October 2013.

[I-D.jiang-config-negotiation-ps]
Jiang, S., Yin, Y., and B. Carpenter, "Network Configuration Negotiation Problem Statement and Requirements", draft-jiang-config-negotiation-ps-02 (work in progress), January 2014.

[RFC1157] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol (SNMP)", STD 15, RFC 1157, May 1990.

[RFC1661] Simpson, W., "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, July 1994.

[RFC1994] Simpson, W., "PPP Challenge Handshake Authentication Protocol (CHAP)", RFC 1994, August 1996.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC2866] Rigney, C., "RADIUS Accounting", RFC 2866, June 2000.

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

[RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.

[RFC4322] Richardson, M. and D. Redelmeier, "Opportunistic Encryption using the Internet Key Exchange (IKE)", RFC 4322, December 2005.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC5575] Marques, P., Sheth, N., Raszuk, R., Greene, B., Mauch, J., and D. McPherson, "Dissemination of Flow Specification Rules", RFC 5575, August 2009.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

Authors' Addresses

Michael H. Behringer
Cisco Systems

Email: mbehring@cisco.com

Brian Carpenter
Department of Computer Science
University of Auckland
PB 92019
Auckland 1142
New Zealand

Email: brian.e.carpenter@gmail.com

Sheng Jiang
Huawei Technologies Co., Ltd
Q14, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: jiangsheng@huawei.com