General Gap Analysis for Autonomic Networking

Abstract

   This document provides a problem statement and general gap analysis
   for an IP-based Autonomic Network that is mainly based on distributed
   network devices. The document provides background by reviewing the
   current status of autonomic aspects of IP networks and the extent to
   which current network management depends on centralization and human
   administrators. Finally, the document outlines the general features
   that are missing from current network abilities and are needed in the
   ideal Autonomic Network concept.

   This document is a product of the IRTF's Network Management Research
   Group.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

    The general goals and relevant definitions for Autonomic Networking
    are discussed in [RFC7575]. 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 centralization. In essence, Autonomic
    Networking is putting all network configurations 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 complex
    troubleshooting, will remain to be done by humans.

    This document represents the consensus of the IRTF's Network
    Management Research Group (NMRG). It first provides background by
    identifying examples of partial autonomic behavior in the Internet
    and by describing important areas of non-autonomic behavior. Based
    on these observations, it then describes missing general mechanisms
    that would allow autonomic behaviors to be added throughout the
    Internet.

2. Terminology

    The terminology defined in [RFC7575] is used in this document.

3. Automatic and Autonomic Aspects of Current IP Networks

    This section discusses the history and current status of automatic or
    autonomic operations in various aspects of network configuration, in
    order to establish a baseline for the gap analysis. In particular,
    routing protocols already contain elements of autonomic processes,
    such as information exchange and state synchronization.

3.1. IP Address Management and DNS

    For many years, there was no alternative to completely manual and
    static management of IP addresses and their prefixes. 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). Even so, prefixes had to
be manually assigned to subnets and their routers, and DHCP servers
had to be configured accordingly.

In terms of management, there is a linkage between IP address
management and DNS management, because DNS mappings typically need to
be appropriately synchronized with IP address assignments. 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 synthesize DNS names even for hosts that only played the
role of clients. Therefore, it became necessary to synchronize DHCP
server tables with forward and reverse DNS. For this reason, IP
address management tools emerged, as discussed for the case of
renumbering in [RFC7010]. These are, however, centralized solutions
that do not exhibit autonomic properties as defined in [RFC7575].

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 it requires
specific configuration so cannot be considered autonomic. How this
topic is to be handled in home networks is still in discussion
[Pfister]. Still further away is autonomic assignment and delegation
of routable IPv4 subnet prefixes.

An IPv6 network needs several aspects of host address assignments to
be configured. The network might use stateless address
autoconfiguration [RFC4862] or DHCPv6 [RFC3315] in stateless or
stateful modes, and there are various alternative forms of Interface
Identifier [RFC7136].

Another feature is the possibility of Dynamic DNS Update [RFC2136].
With appropriate security, this is an automatic approach, where no
human intervention is required to create the DNS records for a host
after it acquires a new address. However, there are coexistence
issues with a traditional DNS setup, as described in [RFC7010].

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 from
   the start, as it was clear that manual configuration of routing
   tables for a large network was impractical.

   IGP routers do need some initial configuration data to start up the
   autonomic routing protocol. Also, BGP-4 routers need detailed static
   configuration of routing policy data.

3.3. Configuration of Default Router in a Host

   Originally, the configuration of a default router in a host was a
   manual operation. Since the deployment of DHCP, this has been
   automatic as far as most IPv4 hosts 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, DHCP 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 essentially autonomic as far as
   all IPv6 hosts are concerned, and DHCPv6 is not involved. However,
   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 (see, for example, [RFC6418]).

3.4. Hostname Lookup

   Originally, hostnames were looked up in a static table, often
   referred to as "hosts.txt" from its traditional file name. 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, needs to be

configured with the appropriate DNS server addresses. Additionally,
some networks deploy Multicast DNS [RFC6762] locally to provide
additional automation of the name space.

3.5. User Authentication and Accounting

Originally, user authentication and accounting was mainly based on
physical connectivity and the degree of trust that follows from
direct connectivity. Network operators charged based on the setup of
dedicated physical links with users. Automated user authentication
was introduced by the Point-to-Point Protocol [RFC1661], [RFC1994]
and RADIUS protocol [RFC2865] [RFC2866] in the early 1990s. As long
as a user completes online authentication through the RADIUS
protocol, the accounting for that user starts on the corresponding
Authentication, Authorization, and Accounting (AAA) server
automatically. This mechanism enables business models with charging
based on the amount of traffic or time. However, user authentication
information continues to be manually managed by network
administrators. It also becomes complex in the case of mobile users
who roam between operators, since prior relationships between the
operators are needed.

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 be applied strictly for all security
configurations. As a result, security has largely been based on
centrally imposed configurations.

Many aspects of security depend on policy, for example, password
rules, privacy rules, firewall rulesets, intrusion detection and
prevention settings, VPN configurations, and the choice of
cryptographic algorithms. Policies are, by definition, human made
and will therefore also persist in an autonomic environment.
However, policies are becoming more high-level, abstracting
addressing, for example, 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 IEEE 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 behaviors. In this case, a security policy
about algorithm strength can be configured into servers but will
propagate automatically to clients.

3.7. State Synchronization

Another area where autonomic processes between peers are involved is
state synchronization. In this case, several devices start out with
inconsistent state and go through a peer-to-peer procedure after
which their states are consistent. Many autonomic or automatic
processes include some degree of implicit state synchronization.
Network time synchronization [RFC5905] is a well-established explicit
example, guaranteeing that a participating node's clock state is
synchronized with reliable time servers within a defined margin of
error, without any overall point of control of the synchronization
process.

4. Current Non-autonomic Behaviors

In current networks, many operations are still heavily dependent on
human intelligence and decision, or on centralized top-down network
management systems. These operations are the targets of Autonomic
Networking technologies. The ultimate goal of Autonomic Networking
is to replace human and automated operations by autonomic functions,
so that the networks can run independently without depending on a
human or Network Management System (NMS) for routine details, while
maintaining central control where required. Of course, there would
still be the absolute minimum of human input required, particularly
during the network-building stage, emergencies, and difficult
troubleshooting.

This section analyzes the existing human and central dependencies in
typical networks and suggests cases where they could, in principle,
be replaced by autonomic behaviors.

4.1. Building a New Network

Building a network requires the operator to analyze the requirements
of the new network, design a deployment architecture 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 central 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 foreseeable future. However, part of these jobs may be able to become autonomic, such as detailed 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 and Management

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 foreseeable future. User identification integration/consolidation among networks or network services is another challenge for Autonomic Network access. Currently, many 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 handle the configuration of network devices. Tools have been developed to enable remote management and make such management easier. However, the decision about each configuration detail depends either on human intelligence or rigid templates. Therefore, these are the sources of all network configuration errors -- the human was wrong, the template was wrong, or both were wrong. This is also a barrier to increasing the utility of network resources, because the human managers cannot respond quickly enough to network events, such as traffic bursts, that were not foreseen in the template. For example, currently, a light load is often assumed in network design because there is no mechanism to properly handle a sudden traffic flood. It is therefore common to avoid performance collapses caused by traffic overload by configuring idle resources, with an overprovisioning ratio of at least 2 being normal [Xiao02].

There are grounds for concern that the introduction of new, more
flexible, methods of network configuration, typified by Software-
Defined Networking (SDN), will only make the management problem more
complex unless the details are managed automatically or
autonomically. There is no doubt that SDN creates both the necessity
and the opportunity for automation of configuration management, e.g.,
[Kim13]. This topic is discussed from a service provider viewpoint
in [RFC7149].

Autonomic decision processes for 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; this would prevent
configuration errors from having lasting impact.

4.3. Security Setup

Setting up security for a network generally requires very detailed
human intervention or relies entirely on default configurations that
may be too strict or too risky for the particular situation of the
network. While some aspects of security are intrinsically top-down
in nature (e.g., broadcasting a specific security policy to all
hosts), others could be self-managed within the network.

In an Autonomic Network, where nodes within a domain have a mutually
verifiable domain identity, security processes could run entirely
automatically. Nodes could identify each other securely, negotiating
required security settings and even shared keys if needed. The
locations of the trust anchors (certificate authority, registration
authority), certificate revocation lists, policy server, etc., can be
found by service discovery. Transactions such as a download of a
certificate revocation list can be authenticated via a common trust
anchor. Policy distribution can also be entirely automated and
secured via a common trust anchor.

These concepts lead to a network where the intrinsic security is
automatic and applied by default, i.e., a "self-protecting" network.
For further discussion, see [Behringer].

4.4. Troubleshooting and Recovery

Current networks suffer difficulties in locating the cause of network
failures. Although network devices may issue many warnings while
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. During a
major failure, many different devices will issue multiple warnings
within a short time, causing overload for the NMS and the operators.

However, for many scenarios, human experience is still vital to
identify real issues and locate them. This situation may be improved
by automatically 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 (particularly hardware
failures) may always require human intervention. However, Autonomic
Network management behavior may help to reduce the impact of errors,
for example, by switching traffic flows around. Today, this is
usually manual (except for classical routing updates). Fixing
software failures and configuration errors currently depends on
humans and may even involve rolling back software versions and
rebooting hardware. 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.

Another possible autonomic function is predicting device failures or
overloads before they occur. A device could predict its own failure
and warn its neighbors, or a device could predict its neighbor's
failure. In either case, an Autonomic Network could respond as if
the failure had already occurred by routing around the problem and
reporting the failure, with no disturbance to users. The criteria
for predicting failure could be temperature, battery status, bit
error rates, etc. The criteria for predicting overload could be
increasing load factor, latency, jitter, congestion loss, etc.

5. Features Needed by Autonomic Networks

   There are innumerable properties of network devices and end systems
   that today need to be configured either manually, by scripting, or by
   using a management protocol such as the Network Configuration
   Protocol (NETCONF) [RFC6241]. In an 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,
   and NTP configuration, but the list is endless.

   The task of Autonomic Networking is to incrementally build up
   individual autonomic processes that could progressively be combined
   to respond to every type of network event. Building on the preceding
   background information, and on the reference model in [RFC7575], this
   section outlines the gaps and missing features in general terms and,
   in some cases, mentions general design principles that should apply.

5.1. More Coordination among Devices or Network Partitions

Network services are dependent on a number of devices and parameters
to be in place in a certain order. For example, after a power
failure, a coordinated sequence of "return to normal" operations is
desirable (e.g., switches and routers first, DNS servers second,
etc.). Today, the correct sequence of events is either known only by
a human administrator or automated in a central script. In a truly
Autonomic Network, elements should understand their dependencies and
be able to resolve them locally.

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. Devices
must be able to derive, for themselves, the dependencies between such
information and configurations.

There are therefore increased requirements for horizontal information
exchange in the networks. Particularly, three types of interaction
among peer network devices are needed for autonomic decisions:
discovery (to find neighbors and peers), synchronization (to agree on
network status), and negotiation (when things need to be changed).
Thus, there is a need for reusable discovery, synchronization, and
negotiation mechanisms that would support the discovery of many
different types of device, the synchronization of many types of
parameter, and the negotiation of many different types of objective.

5.2. Reusable Common Components

Elements of autonomic functions already exist today, within many
different protocols. However, all such functions have their own
discovery, transport, messaging, and security mechanisms as well as
non-autonomic management interfaces. Each protocol has its own
version of the above-mentioned functions to serve specific and narrow
purposes. It is often difficult to extend an existing protocol to
serve different purposes. Therefore, in order to provide the
reusable discovery, synchronization, and negotiation mechanisms
mentioned above, it is desirable to develop a set of reusable common
protocol components for Autonomic Networking. These components
should be:

o Able to identify other devices, users, and processes securely.

o Able to automatically secure operations, based on the above
   identity scheme.

o Able to manage any type of information and information flows.

   o Able to discover peer devices and services for various Autonomic
      Service Agents (or autonomic functions).

   o Able to support closed-loop operations when needed to provide
      self-managing functions involving more than one device.

   o Separable from the specific Autonomic Service Agents (or autonomic
      functions).

   o Reusable by other autonomic functions.

5.3. Secure Control Plane

   The common components will, in effect, act as a control plane for
   autonomic operations. This control plane might be implemented in-
   band as functions of the target network, in an overlay network, or
   even out-of-band in a separate network. Autonomic operations will be
   capable of changing how the network operates and allocating resources
   without human intervention or knowledge, so it is essential that they
   are secure. Therefore, the control plane must be designed to be
   secure against forged autonomic operations and man-in-the middle
   attacks and as secure as reasonably possible against denial-of-
   service attacks. It must be decided whether the control plane needs
   to be resistant to unwanted monitoring, i.e., whether encryption is
   required.

5.4. Less Configuration

   Many existing protocols have been defined to be as flexible as
   possible. Consequently, these protocols need numerous initial
   configurations to start operations. There are choices and options
   that are irrelevant in any particular case, some of which target
   corner cases. Furthermore, in protocols that have existed for years,
   some design considerations are no longer relevant, since the
   underlying hardware technologies have evolved meanwhile. To
   appreciate the scale of this problem, consider that more than 160
   DHCP options have been defined for IPv4. Even sample router
   configuration files readily available online contain more than 200
   lines of commands. There is therefore considerable scope for
   simplifying the operational tools for configuration of common
   protocols, even if the underlying protocols themselves cannot be
   simplified.

   From another perspective, the deep reason why human decisions are
   often needed mainly results from the lack of information. When a
   device can collect enough information horizontally from other
   devices, it should be able to decide many parameters by itself,
   instead of receiving them from top-down configuration.

It is desired that top-down management is reduced in Autonomic
Networking. Ideally, only the abstract Intent is needed from the
human administrators. Neither users nor administrators should need
to create and maintain detailed policies and profiles; if they are
needed, they should be built autonomically. The local parameters
should be decided by distributed Autonomic Nodes themselves, either
from historic knowledge, analytics of current conditions, closed
logical decision loops, or a combination of all.

5.5. Forecasting and Dry Runs

In a conventional network, there is no mechanism for trying something
out safely, which means that configuration changes have to be
designed in the abstract and their probable effects have to be
estimated theoretically. In principle, an alternative to this would
be to test the changes on a complete and realistic network simulator.
However, this is a practical impossibility for a large network that
is constantly changing, even if an accurate simulation could be
performed. There is therefore a risk that applying changes to a
running network will cause a failure of some kind. An autonomic
network could fill this gap by supporting a closed loop "dry run"
mode in which each configuration change could be tested out
dynamically in the control plane without actually affecting the data
plane. If the results are satisfactory, the change could be made
live on the running network. If there is a consistency problem such
as overcommitment of resources or incompatibility with another
configuration setting, the change could be rolled back dynamically
with no impact on traffic or users.

5.6. Benefit from Knowledge

The more knowledge and experience we have, the better decisions we
can make. It is the same for networks and network management. When
one component in the network lacks knowledge that affects what it
should do, and another component has that knowledge, we usually rely
on a human operator or a centralized management tool to convey the
knowledge.

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

However, historic knowledge is very helpful to make correct
decisions, in particular, to reduce 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 relationships between network events
and network configuration may help a network to decide the best
parameters according to real performance feedback.

In addition to such historic knowledge, powerful data analytics of
current network conditions may also be a valuable source of knowledge
that can be exploited directly by Autonomic Nodes.

6. Security Considerations

This document is focused on what is missing to allow autonomic
network configuration, including security settings, of course.
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. Informative References

[Behringer]
           Behringer, M., Pritikin, M., and S. Bjarnason, "Making The
           Internet Secure By Default", Work in Progress,
           draft-behringer-default-secure-00, January 2014.

[Kim13]   Kim, H. and N. Feamster, "Improving Network Management
           with Software Defined Networking", IEEE Communications
           Magazine, pages 114-119, February 2013.

[Pfister] Pfister, P., Paterson, B., and J. Arkko, "Distributed
           Prefix Assignment Algorithm", Work in Progress,
           draft-ietf-homenet-prefix-assignment-07, June 2015.

[RFC1661] Simpson, W., Ed., "The Point-to-Point Protocol (PPP)",
           STD 51, RFC 1661, DOI 10.17487/RFC1661, July 1994,
           <http://www.rfc-editor.org/info/rfc1661>.

[RFC1994] Simpson, W., "PPP Challenge Handshake Authentication
           Protocol (CHAP)", RFC 1994, DOI 10.17487/RFC1994, August
           1996, <http://www.rfc-editor.org/info/rfc1994>.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol",
           RFC 2131, DOI 10.17487/RFC2131, March 1997,
           <http://www.rfc-editor.org/info/rfc2131>.

[RFC2136] Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound,
           "Dynamic Updates in the Domain Name System (DNS UPDATE)",
           RFC 2136, DOI 10.17487/RFC2136, April 1997,
           <http://www.rfc-editor.org/info/rfc2136>.

   [RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson,
              "Remote Authentication Dial In User Service (RADIUS)",
              RFC 2865, DOI 10.17487/RFC2865, June 2000,
              <http://www.rfc-editor.org/info/rfc2865>.

   [RFC2866] Rigney, C., "RADIUS Accounting", RFC 2866,
              DOI 10.17487/RFC2866, June 2000,
              <http://www.rfc-editor.org/info/rfc2866>.

   [RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins,
              C., and M. Carney, "Dynamic Host Configuration Protocol
              for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July
              2003, <http://www.rfc-editor.org/info/rfc3315>.

   [RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic
              Host Configuration Protocol (DHCP) version 6", RFC 3633,
              DOI 10.17487/RFC3633, December 2003,
              <http://www.rfc-editor.org/info/rfc3633>.

   [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman,
              "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861,
              DOI 10.17487/RFC4861, September 2007,
              <http://www.rfc-editor.org/info/rfc4861>.

   [RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless
              Address Autoconfiguration", RFC 4862,
              DOI 10.17487/RFC4862, September 2007,
              <http://www.rfc-editor.org/info/rfc4862>.

   [RFC5575] Marques, P., Sheth, N., Raszuk, R., Greene, B., Mauch, J.,
              and D. McPherson, "Dissemination of Flow Specification
              Rules", RFC 5575, DOI 10.17487/RFC5575, August 2009,
              <http://www.rfc-editor.org/info/rfc5575>.

   [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch,
              "Network Time Protocol Version 4: Protocol and Algorithms
              Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010,
              <http://www.rfc-editor.org/info/rfc5905>.

   [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed.,
              and A. Bierman, Ed., "Network Configuration Protocol
              (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011,
              <http://www.rfc-editor.org/info/rfc6241>.

   [RFC6418] Blanchet, M. and P. Seite, "Multiple Interfaces and
              Provisioning Domains Problem Statement", RFC 6418,
              DOI 10.17487/RFC6418, November 2011,
              <http://www.rfc-editor.org/info/rfc6418>.

   [RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762,
              DOI 10.17487/RFC6762, February 2013,
              <http://www.rfc-editor.org/info/rfc6762>.

   [RFC7010] Liu, B., Jiang, S., Carpenter, B., Venaas, S., and W.
              George, "IPv6 Site Renumbering Gap Analysis", RFC 7010,
              DOI 10.17487/RFC7010, September 2013,
              <http://www.rfc-editor.org/info/rfc7010>.

   [RFC7136] Carpenter, B. and S. Jiang, "Significance of IPv6
              Interface Identifiers", RFC 7136, DOI 10.17487/RFC7136,
              February 2014, <http://www.rfc-editor.org/info/rfc7136>.

   [RFC7149] Boucadair, M. and C. Jacquenet, "Software-Defined
              Networking: A Perspective from within a Service Provider
              Environment", RFC 7149, DOI 10.17487/RFC7149, March 2014,
              <http://www.rfc-editor.org/info/rfc7149>.

   [RFC7575] Behringer, M., Pritikin, M., Bjarnason, S., Clemm, A.,
              Carpenter, B., Jiang, S., and L. Ciavaglia, "Autonomic
              Networking: Definitions and Design Goals", RFC 7575,
              DOI 10.17487/RFC7575, June 2015,
              <http://www.rfc-editor.org/info/rfc7575>.

   [Xiao02]   Xiao, X., Telkamp, T., Fineberg, V., Chen, C., and L. Ni,
              "A Practical Approach for Providing QoS in the Internet
              Backbone", IEEE Communications Magazine, pages 56-62,
              December 2002.

Authors' Addresses

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

   EMail: jiangsheng@huawei.com

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

   EMail: brian.e.carpenter@gmail.com

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

   EMail: mbehring@cisco.com