Network Configuration Negotiation Problem Statement and Requirements
draft-jiang-config-negotiation-ps-01

Abstract

   This document describes a problem statement and general requirements
   for distributed autonomous configuration of multiple aspects of
   networks, in particular carrier networks. The basic model is that
   network elements need to negotiate configuration settings with each
   other to meet overall goals. The document describes a generic
   negotiation behavior model. The document also reviews whether
   existing management and configuration protocols may be suitable for
   autonomic networks.

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 April 11, 2014.

Table of Contents

1. Introduction

    The success of IP and the Internet has made the network model very
    complicated, and networks have become larger and larger. The network
    of a large ISP typically contains more than a hundred thousand
    network devices which play many roles. The initial setup
    configuration, dynamic management and maintenance, troubleshooting
    and recovery of these devices have become a huge outlay for network
    operators. Particularly, these devices are managed by many different
    staff requiring very detailed training and skills. The coordination
    of these staff is also difficult and often inefficient. There are
    therefore increased requirements for autonomy in the networks.
    [I-D.boucadair-network-automation-requirements] is one of the
    attempts to describe such requirements. It listed a "requirement for
    a protocol to convey configuration information towards the managed
    entities". However, the present document is going further by
    requiring a configuration negotiation protocol rather than
    undirectional provisioning.

    Autonomic operation means network devices could decide configurations
    by themselves. There are already many existing internal
    implementations or algorithms for a network device to decide or

compute its configuration according to its own status, often referred to as device intelligence. In one particular area, routing protocols, distributed autonomous configuration is a well established mechanism. The question is how to extend autonomy to cover all kinds of configuration, not just routing tables.

However, in order to make right or good decisions, the network devices need to know more information than just routes from the relevant or neighbor devices. There are dependencies between such information and configurations. Currently, most of these configurations require manual coordination and operation.

Today, there is no generic negotiation protocol that can be used to control decision processes among distributed devices or between networks. Proprietary network management systems are widely used but tend to be hierarchical systems ultimately relying on a console operator and a central database. An autonomous system needs to be less hierarchical and with less dependence on an operator. This requires network elements to negotiate directly with each other, with an absolute minimum or zero configuration data at the installation stage.

This document analyzes the requirements for a generic negotiation protocol and the application scenarios, then gives considerations for detailed technical requirements for designing such a protocol. Some existing protocols are also reviewed as part of the analysis. A protocol behavior model, which may be used to define such a negotiation protocol, is also described.

2. Requirements and Application Scenarios for Network Devices Negotiation

Routing protocols are a typical autonomic model based on distributed devices. But routing is mainly one-way information announcement (in both directions), rather than bi-directional negotiation. Its only focus is reachability. The future networks need to be able to manage many more dimensions of the network, such as power saving, load balancing, etc. The current routing protocols only show simple link status, as up or down. More information, such as latency, congestion, capacity, and particularly available throughput, is very helpful to get better path selection and utilization rate.

A negotiation model with no human intervention is needed when the coordination of multiple devices can provide better overall network performance.

A negotiation model provides a possibility for forecasting. A "dry run" becomes possible before the concrete configuration takes place.

Another area is tunnel management, with automatic setup, maintenance, and removal. A related area is ad hoc routes, without encapsulation, to handle specific traffic flows (which might be regarded as a form of software defined networking).

When a new user or device comes online, it might be necessary to set up resources on multiple relevant devices, coordinated and matched to each other so that there is no wasted resource. Security settings might also be needed to allow for the new user/device.

Status information and traffic metrics need to be shared between nodes for dynamic adjustment of resources.

Troubleshooting should be as autonomous as possible. Although it is far from trivial, there is a need to detect the "real" breakdown amongst many alerts, and then take action to reconfigure the relevant devices. Again, routing protocols have done this for many years, but in an autonomous network it is not just routing that needs to reconfigure itself after a failure.

2.1. Negotiation between downstream and upstream network devices

The typical scenario is that there is a new access gateway, which could be a wireless base station, WiFi hot spot, Data Center switch, VPN site switch, enterprise CE, home gateway, etc. When it is plugged into the network, bi-direction configuration/control is needed. The upstream network needs to configure the device, its delegated prefix(es), DNS server, etc. For this direction, DHCP might be suitable and sufficient. However, there is another direction: the connection of downstream devices also needs to trigger the upstream devices, for example the provider edge, to create a corresponding configuration, by setting up a new tunnel, service, authentication, etc.

Furthermore, after the communication between gateway and provider has been established, the devices would like to optimize their configurations interactively according to dynamic link status or performance measurements, power consumption, etc. For dynamical management and maintenance, there are many other network events that downstream network devices may need to report to upstream network devices and initiate some configuration change on these upstream networks. Currently, these kinds of synchronizing operations require the involvement of human operators.

Similar requirements can also appear between other types of downstream and upstream network devices.

2.2. Negotiation between peer network devices

   Within a large network, in many segments, there are network devices
   that are of equal importance. They have a peer rather than
   hierarchical relationship. There are many horizontal traffic flows
   or tunnels between them. In order to make their connection
   efficient, their configurations have to match each other. Any change
   of their configuration may request synchronization with their peer
   network devices.

   However, in many cases, the peer network devices may not be able to
   make the exact changes as requested. Instead, another slightly
   different change may be the best choice for optimal performance. In
   order to decide on this best choice, multiple rounds of information
   exchange between peers may be necessary. This should be done without
   requiring the involvement of human operators. To provide this
   ability, a mechanism for network devices to be able to negotiate with
   each other is needed.

2.3. Negotiation between networks

   A network may announce some information about its internal
   capabilities to connected peer networks, so that the peer networks
   can react accordingly. BGP routing information is a simple example.

   Beyond reachability, more information may enable better coordination
   among networks. Examples include traffic engineering among multiple
   connections between two networks, particularly when these connections
   are geographically distributed; dynamic bandwidth adjustment to match
   changing traffic from a peer network; dynamic establishment and
   adjustment of differentiated service classes to support Service Level
   Agreements; and so on.

2.4. Information and status query among devices

   In distributed routers, many data such as status indicators or
   traffic measurements are dynamically changing. These may be the
   triggers for follow-up negotiation. For example, consider two
   routers sharing traffic load. Router A may request the traffic
   situation of router B, then start negotiation, such as requesting
   router B to handle all traffic, so that router A can enter power-
   saving mode. Another example is that a device may request its
   neighbor to send a forecast or dry-run result based on a given
   potential configuration change. Then, the initiating router can
   evaluate whether the potential configuration change could meet its
   original target.

2.5. Unavoidable configuration

   Even with autonomous negotiation, some initial configuration data
   cannot be avoided in some devices. A design goal is to reduce this
   to an absolute minimum. This information may have to be pre-
   configured on the device before it has been deployed physically, and
   is typically static. A preliminary list of unavoidable configuration
   data is:

   o Authentic identity for each device. This may be a public key or a
      signed certification. This is necessary to protect the
      infrastructure against unauthorized replacement of equipment.

   o The role/function and capability of the device. The role/function
      may depend on the network planning. The capability is typically
      decided by the hardware.

   o On the network edge, the routers may need to be configured with
      the identity of each peer provider, and their entitlements to
      service.

   Ideally, everything else (topology, link capacity, address prefixes,
   shared resources, customer authentication and authority, etc.) will
   be discovered or negotiated autonomously according to general policy
   for various negotiated objectives.

3. Existing protocols

   Routing protocols are mainly one-way information announcements. The
   receiver makes decisions independently, based on the received
   information, and there is not much feedback information to the
   announcing peer. This remains true even though the protocol is used
   in both directions between peer routers; there is no negotiation, and
   each peer runs its route calculations independently.

   There are many existing protocols that have some minor negotiation
   abilities, such as Dynamic Host Configuration Protocol for IPv6
   (DHCPv6) [RFC3315], Neighbor Discovery (ND) [RFC4861], Port Control
   Protocol (PCP) [RFC6887], etc. Numerous other configuration or
   management protocols could be listed. However, they are either
   simple request/response models or can only negotiate on very limited
   aspects. Negotiation is a feature of certain connectivity protocols,
   for example Point to Point Protocol (PPP) [RFC1661] and Transport
   Layer Security (TLS) [RFC5246] but again, the negotiable aspects are
   strictly limited.

   At present, there appears to be no generic protocol that meets the
   requirements discussed above.

4. A Behavior Model of a Generic Negotiation Protocol

This section describes a behavior model and some considerations for designing a generic negotiation protocol, which would act as a platform for different negotiation objectives.

o A generic platform

   The design of the network device protocol is desired to be a generic platform, which is independent from the negotiation contents. It should only take care of the general intercommunication between negotiation counterparts. The negotiation contents will vary according to the various negotiation objectives and the different pairs of negotiating counterparts.

o Security infrastructure and trust relationship

   Because this negotiation protocol may directly cause changes to device configurations and bring significant impacts to a running network, this protocol must be based on a restrictive security infrastructure. It should be carefully managed and monitored so that every device in this negotiation system behaves well and remains well protected.

   On the other hand, a limited negotiation model might be deployed based on a limited trust relationship. For example, between two administrative domains, devices may also exchange limited information and negotiate some particular configurations based on a limited conventional or contractual trust relationship.

o The uniform pattern for negotiation contents

   The negotiation contents should be defined according to a uniform pattern. They could be carried either in TLV (Type, Length and Value) format or in payloads described by a flexible language, like XML. A protocol design should choose one of these two. The format must be extensible for unknown future requirements.

o A simple initiator/responder model

   Multiple-party negotiations are too complicated to be modeled and there may be too many dependencies among the parties to converge efficiently. A simple initiator/responder model is more feasible and could actually complete multiple-party negotiations by indirect steps. Naturally this process must be guaranteed to terminate and must contain tie-breaking rules.

o Organizing of negotiation content

   Naturally, the negotiation content should be organized according
   to the relevant function or service. The content from different
   functions or services should be kept independent from each other.
   They should not be combined into a single option or single session
   because these contents may be negotiated with different
   counterparts or may be different in response time.

o Topology neighbor device discovery

   Every network device that supports the negotiation protocol is a
   responder and always listens to a well-known (UDP?) port. A well-
   known link-local multicast address should be defined for discovery
   purposes. Upon receiving a discovery or request message, the
   recipient device should return a message in which it either
   indicates itself as a proper negotiation counterpart or diverts
   the initiator towards another more proper device.

o Self aware network devices

   Every network device should be pre-configured with its role and
   functions and be aware of its own capabilities. The roles may be
   only distinguished because of network behaviors, which may include
   forwarding behaviors, aggregation properties, topology location,
   bandwidth, tunnel or translation properties, etc. The role and
   function may depend on the network planning. The capability is
   typically decided by the hardware or firmware. It is the
   foundation of the negotiation behavior of a specific device.

o Requests and responses in negotiation procedures

   The initiator should be able to negotiate with its relevant
   negotiation counterpart devices, which may be different according
   to the negotiation objective. It may request relevant information
   from the negotiation counterpart so that it can decide its local
   configuration to give the most coordinated performance. It may
   request the he negotiation counterpart to make a matching
   configuration in order to set up a successful communication with
   it. It may request certain simulation or forecast results by
   sending some dry run conditions.

   Beyond the traditional yes/no answer, the responder should be able
   to reply with a suggested alternative if its answer is 'no'. This
   is going to start a bi-direction negotiation towards reaching a
   compromise between the two devices.

o Convergence of negotiation procedures

The negotiation procedure should move towards convergent results.
It means that when a responder makes a suggestion of a changed
condition in a negative reply, it should be as close as possible
to the original request or previous suggestion. The suggested
value of the third or later negotiation steps should be chosen
between the suggested values from the last two negotiation steps.
In any case there must be a mechanism to guarantee rapid
convergence in a small number of steps.

o Dependencies of negotiation

   In order to decide a configuration on a device, the router may
   need information from neighbor routers. This can be reached
   through the above negotiation procedure. However, a certain
   information on a neighbor router may depend on other information
   from its neighbors, which may need another negotiation procedure
   to obtain or decide. Therefore, there are dependencies among
   negotiation procedures. There need to be clear edge/convergence
   for these negotiation dependencies. Also some mechanisms are
   needed to avoid loop dependencies.

o End of negotiation

   A single negotiation procedure also needs ending conditions if it
   does not converge. A limit round number, for example three,
   should be set on the devices. It may be an implementation choice
   or a configurable parameter. However, the protocol design needs
   to clearly specify this, so that the negotiation can be terminated
   properly. In some cases, a tiemout might be needed to break off a
   dependency negotiation.

o Failed negotiation

   There must be a well-defined procedure for concluding that a
   negotiation cannot succeed, and if so deciding what happens next
   (deadlock resolution, tie-breaking, or revert to best effort
   service).

o Policy constraints

   There must be provision for general policy rules to be applied by
   all devices in the network (e.g., security rules, prefix length,
   resource sharing rules). However, policy distribution might not
   use the negotiation protocol itself.

o Management monitoring, alerts and intervention

Devices should be able to report to a monitoring system. Some
events must be able to generate operator alerts and some provision
for emergency intervention must be possible (e.g. to freeze
negotiation in a mis-behaving device). These features may not use
the negotiation protocol itself.

[5]. Security Considerations

This document does not include a detailed threat analysis for
autonomous configuration, but it is obvious that a successful attack
on autonomic nodes would be extremely harmful, as such nodes might
end up with a completely undesirable configuration. A concrete
protocol proposal will therefore require a threat analysis, and some
form of strong authentication and, if possible, built-in protection
against denial of service attacks.

Separation of network devices and user devices may become very
helpful in this kind of scenario.

Also, security configuration itself should become autonomic whenever
possible. However, in the security area at least, operator override
of autonomic configuration must be possible for emergency use.

As noted earlier, a cryptographically authenticated identity for each
device is needed in an autonomic network. It is not safe to assume
that a large network is physically secured against interference or
that all personnel are trustworthy. Each autonomous device should be
capable of proving its identity and authenticating its messages. One
approach would be to use a private/public key pair and sufficiently
strong cryptography. Each device would generate its own private key,
which is never exported from the device. The device identity and
public key would be recorded in a network-wide database. The
alternative of using symmetric keys (shared secrets) is less
attractive, since it creates a risk of key leakage as well as a key
management problem when devices are installed or removed.

Generally speaking, no personal information is expected to be
involved in the negotiation protocol, so there should be no direct
impact on personal privacy. Nevertheless, traffic flow paths, VPNs,
etc. may be negotiated, which could be of interest for traffic
analysis. Also, carriers generally want to conceal details of their
network topology and traffic density from outsiders. Therefore,
since insider attacks cannot be prevented in a large carrier network,
the security mechanism for the negotiation protocol needs to provide
message confidentiality.

[6]. IANA Considerations

This draft does not request any IANA action.

7. Acknowledgements

The authors want to thank Zhenbin Li, Bing Liu for valuable comments.

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

8. Change Log [RFC Editor please remove]

draft-jiang-negotiation-config-ps-01, add more requirements, and add more considerations for behavior model, 2013-10-08.

draft-jiang-negotiation-config-ps-00, original version, 2013-06-29.

9. Informative References

[I-D.boucadair-network-automation-requirements]
          Boucadair, M. and C. Jacquenet, "Requirements for
          Automated (Configuration) Management", draft-boucadair-
          network-automation-requirements-01 (work in progress),
          June 2013.

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

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

[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.

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

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security
          (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC6887] Wing, D., Cheshire, S., Boucadair, M., Penno, R., and P.
          Selkirk, "Port Control Protocol (PCP)", RFC 6887, April
          2013.

Authors' Addresses

   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

   Yuanbin Yin
   Huawei Technologies Co., Ltd
   Q15, Huawei Campus, No.156 Beiqing Road
   Hai-Dian District, Beijing, 100095
   P.R. China

   Email: yinyuanbin@huawei.com

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

   Email: brian.e.carpenter@gmail.com