ANIMA
Internet-Draft
Intended status: Informational
Expires: December 13, 2015

M. Behringer, Ed.
Cisco
B. Carpenter
Univ. of Auckland
T. Eckert
Cisco
L. Ciavaglia
Alcatel Lucent
B. Liu
Huawei Technologies
June 11, 2015

A Reference Model for Autonomic Networking
draft-behringer-anima-reference-model-02

Abstract

This document describes a reference model for Autonomic Networking. The goal is to define how the various elements in an autonomic context work together, to describe their interfaces and relations. While the document is written as generally as possible, the initial solutions are limited to the chartered scope of the WG.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The document "Autonomic Networking - Definitions and Design Goals" [I-D.irtf-nmrg-autonomic-network-definitions] explains the fundamental concepts behind Autonomic Networking, and defines the relevant terms in this space. In section 5 it describes a high level reference model. This document defines this reference model with more detail, to allow for functional and protocol specifications to be developed in an architecturally consistent, non-overlapping manner. While the document is written as generally as possible, the initial solutions are limited to the chartered scope of the WG.

As discussed in [I-D.irtf-nmrg-autonomic-network-definitions], the goal of this work is not to focus exclusively on fully autonomic nodes or networks. In reality, most networks will run with some autonomic functions, while the rest of the network is traditionally managed. This reference model allows for this hybrid approach.

This is a living document and will evolve with the technical solutions developed in the ANIMA WG. Sections marked with (*) do not represent current charter items. While this document must give a long term architectural view, not all functions will be standardized at the same time.

2. The Network View

This section describes the various elements in a network with autonomic functions, and how these entities work together, on a high level. Subsequent sections explain the detailed inside view for each of the autonomic network elements, as well as the network functions (or interfaces) between those elements.

Figure 1 shows the high level view of an Autonomic Network. It consists of a number of autonomic nodes, which interact directly with each other. Those autonomic nodes provide a common set of capabilities across the network, called the "Autonomic Networking Infrastructure" (ANI). The ANI provides functions like naming, addressing, negotiation, synchronization, discovery and messaging.

Autonomic functions typically span several, possibly all nodes in the network. The atomic entities of an autonomic function are called the "Autonomic Service Agents" (ASA), which are instantiated on nodes.

```
+- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - +
:            :       Autonomic Function 1        :                  :
: ASA 1      :      ASA 1      :      ASA 1      :          ASA 1  :
+- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - +
             :                 :                 :
             :   +- - - - - - - - - - - - - - +  :
             :   :   Autonomic Function 2     :  :
             :   :  ASA 2      :      ASA 2   :  :
             :   +- - - - - - - - - - - - - - +  :
             :                 :                 :
+- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - +
:                Autonomic Networking Infrastructure                :
+- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - +
+--------+   :    +--------+   :    +--------+   :        +--------+
| Node 1 |--------| Node 2 |--------| Node 3 |----...-----| Node n |
+--------+   :    +--------+   :    +--------+   :        +--------+
```

Figure 1: High level view of an Autonomic Network

In a horizontal view, autonomic functions span across the network, as well as the Autonomic Networking Infrastructure. In a vertical view, a node always implements the ANI, plus it may have one or several Autonomic Service Agents.

The Autonomic Networking Infrastructure (ANI) therefore is the foundation for autonomic functions. The current charter of the ANIMA WG is to specify the ANI, using a few autonomic functions as use cases.

3. The Autonomic Network Element

3.1. Architecture

This section describes an autonomic network element and its internal architecture. The reference model explained in [I-D.irtf-nmrg-autonomic-network-definitions] shows the sources of information that an autonomic service agent can leverage: Self-knowledge, network knowledge (through discovery), Intent, and feedback loops. Fundamentally, there are two levels inside an autonomic node: the level of Autonomic Service Agents, and the level of the Autonomic Networking Infrastructure, with the former using the services of the latter. Figure 2 illustrates this concept.

```
   +------------------------------------------------------------+
   |                                                            |
   | +-----------+        +------------+        +------------+  |
   | | Autonomic |        | Autonomic  |        | Autonomic  |  |
   | | Service   |        | Service    |        | Service    |  |
   | | Agent 1   |        | Agent 2    |        | Agent 3    |  |
   | +-----------+        +------------+        +------------+  |
   |       ^                    ^                     ^         |
   | -  -  | -  - API level -  -| -  -  -  -  -  -  - |-  -  -  |
   |       V                    V                     V         |
   |------------------------------------------------------------|
   | Autonomic Networking Infrastructure                        |
   |    - Data structures (ex: certificates, peer information)  |
   |    - Autonomic Control Plane                               |
   |    - discovery, negotiation and synchronisation functions  |
   |    - Intent distribution                                   |
   |    - aggregated reporting and feedback loops               |
   |    - routing                                               |
   |------------------------------------------------------------|
   |             Basic Operating System Functions               |
   +------------------------------------------------------------+
```

Figure 2: Model of an autonomic node

The Autonomic Networking Infrastructure (lower part of Figure 2) contains node specific data structures, for example trust information about itself and its peers, as well as a generic set of functions, independent of a particular usage. This infrastructure should be generic, and support a variety of Autonomic Service Agents (upper part of Figure 2). The Autonomic Control Plane is the summary of all interactions of the Autonomic Networking Infrastructure with other nodes and services.

The use cases of "Autonomics" such as self-management, self-optimisation, etc, are implemented as Autonomic Service Agents. They use the services and data structures of the underlying autonomic networking infrastructure. The underlying Autonomic Networking Infrastructure should itself be self-managing.

The "Basic Operating System Functions" include the "normal OS", including the network stack, security functions, etc.

3.2. Unconstrained AN Nodes

Unconstrained nodes have the full ANI, with the full functionality (details to be worked out). They support all the capabilities outlined in the rest of the document. [tbc]

3.3. Constrained AN Nodes (*)

Constrained nodes have a reduced ANI, with a well-defined minimal functionality (details to be worked out): They do need to be able to join the network, and communicate with at least a helper node which has full ANI functionality. Capabilities of constrained nodes need to be defined here. [tbc]

4. The Autonomic Networking Infrastructure

The Autonomic Networking Infrastructure provides a layer of common functionality across an Autonomic Network. It comprises "must implement" functions and services, as well as extensions.

An Autonomic Function, comprising of Autonomic Service Agents on nodes, can rely on the fact that all nodes in the network implement at least the "must implement" functions.

4.1. Naming

Inside a domain, each autonomic device needs a domain specific identifier. [tbc]

4.2. Addressing

Autonomic Service Agents (ASAs) need addressing to communicate with each other. This section describes the addressing approach of the Autonomic Networking Infrastructure, used by ASAs. It does NOT describe addressing approaches for the data plane of the network, which may be configured and managed in the traditional way. ASAs may provide a service to negotiate address space, or addressing mechanisms for the date plane. One use case for such an autonomic function is described in [I-D.jiang-auto-addr-management]. The addressing the ASAs use is in scope for this section, the address space they negotiate for the data plane is not.

It is generally desirable to make the addressing scheme of the Autonomic Networking Infrastructure as self-managing (autonomic) as possible.

This section is currently under discussion. We currently believe that we should address the following questions:

o How addressing inside the Autonomic Control Plane (ACP) [I-D.behringer-anima-autonomic-control-plane] is assigned and managed autonomically.

- Whether, if there is no separated ACP, Autonomic Service Agents (ASAs) shall have their own address space, or whether they should use the address space configured by the administrator.

- How addressing is handled in the presence of non-autonomic nodes.

- Prefix assignment to interfaces.

- Whether links and link interfaces should get routable address space, or whether link local addressing is sufficient.

- How the address space used in the Autonomic Networking Infrastructure is assigned and managed.

It is not clear at this point whether a specific address scheme should be included in this document, or whether this document should only define the requirements. This is for further discussion.

The document [I-D.behringer-anima-autonomic-addressing] describes one way to autonomically assign loopback addresses to autonomic nodes, in an autonomic, self-managed way. Other ideas and suggestions are strongly encouraged.

4.3. Discovery

Traditionally, most of the information a node requires is provided through configuration or northbound interfaces. An autonomic function should only minimally rely on such northbound interfaces, therefore it needs to discover resources in the network. This section describes various discovery functions in an autonomic network.

Discovering nodes and their properties: A core function to establish an autonomic domain is the discovery of autonomic nodes, primarily adjacent nodes. This may either leverage existing neighbour discovery mechanisms, or new mechanisms.

Discovering services: Network services such as AAA should also be discovered and not configured. Service discovery is required for such tasks. An autonomic network can either leverage existing service discovery functions, or build a new approach.

Thus the discovery mechanism could either be fully integrated with negotiation and synchronization (next section) or could use an independent discovery mechanism such as DNS Service Discovery or Service Location Protocol. This choice is made independently for each Autonomic Service Agent.

4.4. Signaling Between Autonomic Nodes

Autonomic nodes must communicate with each other, for example to negotiate and/or synchronize network parameters of any kind and complexity. This requires some form of signaling between autonomic nodes. The document "A Generic Discovery and Negotiation Protocol for Autonomic Networking" [I-D.carpenter-anima-gdn-protocol] describes requirements for negotiation and synchronization in an autonomic network. It also defines a protocol, GDNP, for this purpose, including an integrated but optional discovery protocol.

4.5. Intent Distribution

Intent is the policy language of an Autonomic Network, see Section 7.2 for general information on Intent. The distribution of Intent is also a function of the Autonomic Control Plane. Various methods can be used to distribute Intent across an autonomic domain.

4.6. Routing

All autonomic nodes in a domain must be able to communicate with each other, and with autonomic nodes outside their own domain. Therefore, an Autonomic Control Plane relies on a routing function. For Autonomic Networks to be interoperable, they must all support one common routing protocol.

4.7. The Autonomic Control Plane

The totality of autonomic interactions forms the "Autonomic Control Plane". This control plane can be either implemented in the global routing table of a node, such as IGPs in today's networks; or it can be provided as an overlay network, as described in [I-D.behringer-anima-autonomic-control-plane].

The ACP can be operated in two ways: 1) as a separate virtual overlay network, as described in [I-D.behringer-anima-autonomic-control-plane]. or 2), in the global routing table. This sections discusses implications of both choices.

Also: Need to address how a separated ACP and an inline ACP cooperate.

5. Security and Trust Infrastructure

An Autonomic Network is self-protecting. All protocols are secure by default, without the requirement for the administrator to explicitly configure security.

Autonomic nodes have direct interactions between themselves, which must be secured. Since an autonomic network does not rely on configuration, it is not an option to configure for example pre-shared keys. A trust infrastructure such as a PKI infrastructure must be in place. This section describes the principles of this trust infrastructure.

A completely autonomic way to automatically and securely deploy such a trust infrastructure is to set up a trust anchor for the domain, and then use an approach as in the document "Bootstrapping Key Infrastructures" [I-D.pritikin-bootstrapping-keyinfrastructures].

5.1. Public Key Infrastructure

An autonomic domain uses a PKI model. The root of trust is a certification authority (CA). A registrar acts as a registration authority (RA).

A minimum implementation of an autonomic domain contains one CA, one Registrar, and network elements.

5.2. Domain Certificate

We need to define how the fields in a domain certificate are to be used. [tbc]

5.3. The MASA

Explain briefly the function, point to [I-D.pritikin-bootstrapping-keyinfrastructures]. [tbc]

5.4. Sub-Domains (*)

Explain how sub-domains are handled. (tbc)

5.5. Cross-Domain Functionality (*)

Explain how trust is handled between different domains. (tbc)

6. Autonomic Service Agents (ASA)

This section describes how autonomic services run on top of the Autonomic Networking Infrastructure.

6.1. General Description of an ASA

general concepts, such as sitting on top of the ANI, etc. Also needs to explain that on a constrained node (see Section 3.3) not all ASAs may run, so we have two classes of ASAs: Ones that run on an unconstrained node, and limited function ASAs that run also on constrained nodes. We expect unconstrained nodes to support all ASAs.

6.2. Specific ASAs for the Enrolment Process

The following ASAs provide essential, required functionality in an autonomic network, and are therefor mandatory to implement on unconstrained autonomic nodes.

6.2.1. The Enrolment ASA

This section describes the function of an autonomic node to bootstrap into the domain with the help of an enrolment proxy (see previous section). [tbc]

6.2.2. The Enrolment Proxy ASA

This section describes the function of an autonomic node that helps a non-enrolled, adjacent devices to enrol into the domain. [tbc]

6.2.3. The Registrar ASA

This section describes the registrar function in an autonomic network. It explains the tasks of a registrar element, and how registrars are placed in a network, redundancy between several, etc. [tbc]

7. Management and Programmability

This section describes how an Autonomic Network is managed, and programmed.

7.1. How an AN Network Is Managed

Explain co-existence of traditional methods (SNMP, syslog, NETCONF, etc) with new, autonomic methods. Those are: Intent, Aggregated Reporting and feedback loops to the NOC. [tbc]

7.2. Intent (*)

This section describes Intent, and how it is managed. Explaining ingest of intent, distribution, the nature (on top of what's in [I-D.irtf-nmrg-autonomic-network-definitions]). That intent is signed, time stamps, etc. Probably pointing back to [I-D.irtf-nmrg-autonomic-network-definitions]. (Note intent distribution is handled in Section 4.5) [tbc]

7.3. Aggregated Reporting (*)

An autonomic network offers through the autonomic control plane the possibility to aggregate information inside the network, before sending it to the admin of the network. While this can be seen or implemented as a specific form of negotiation, the use case is different and therefore mentioned here explicitly.

7.4. Feedback Loops to NOC(*)

Feedback loops are required in an autonomic network to allow the intervention of a human administrator or central control systems, while maintaining a default behaviour. Through a feedback loop an administrator can be prompted with a default action, and has the possibility to acknowledge or override the proposed default action.

7.5. APIs (*)

Need considerations for APIs: How can ASAs use the ANI? Where do we need APIs? [tbc]

7.6. Data Model (*)

Need considerations for a data model. What it should cover, scope. [tbc]

8. Coordination Between Autonomic Functions (*)

8.1. The Coordination Problem (*)

Different autonomic functions may conflict in setting certain parameters. For example, an energy efficiency function may want to shut down a redundant link, while a load balancing function would not want that to happen. The administrator must be able to understand and resolve such interactions, to steer autonomic network performance to a given (intended) operational point.

Several interaction types may exist among autonomic functions, for example:

- Cooperation: An autonomic function can improve the behavior or performance of another autonomic function, such as a traffic forecasting function used by a traffic allocation function.

- Dependency: An autonomic function cannot work without another one being present or accessible in the autonomic network.

- Conflict: A metric value conflict is a conflict where one metric is influenced by parameters of different autonomic functions. A parameter value conflict is a conflict where one parameter is modified by different autonomic functions.

Solving the coordination problem beyond one-by-one cases can rapidly become intractable for large networks. Specifying a common functional block on coordination is a first step to address the problem in a systemic way. The coordination life-cycle consists in three states:

- At build-time, a "static interaction map" can be constructed on the relationship of functions and attributes. This map can be used to (pre-)define policies and priorities on identified conflicts.

- At deploy-time, autonomic functions are not yet active/acting on the network. A "dynamic interaction map" is created for each instance of each autonomic functions and on a per resource basis, including the actions performed and their relationships. This map provides the basis to identify conflicts that will happen at run-time, categorize them and plan for the appropriate coordination strategies/mechanisms.

- At run-time, when conflicts happen, arbitration is driven by the coordination strategies. Also new dependencies can be observed and inferred, resulting in an update of the dynamic interaction map and adaptation of the coordination strategies and mechanisms.

Multiple coordination strategies and mechanisms exists and can be devised. The set ranges from basic approaches such as random process or token-based process, to approaches based on time separation and hierarchical optimization, to more complex approaches such as multi-objective optimization, and other control theory approaches and algorithms family.

8.2. A Coordination Functional Block (*)

A common coordination functional block is a desirable component of the ANIMA reference model. It provides a means to ensure network properties and predictable performance or behavior such as stability,

and convergence, in the presence of several interacting autonomic functions.

A common coordination function requires:

- o A common description of autonomic functions, their attributes and life-cycle.
- o A common representation of information and knowledge (e.g., interaction maps).
- o A common "control/command" interface between the coordination "agent" and the autonomic functions.

Guidelines, recommendations or BCPs can also be provided for aspects pertaining to the coordination strategies and mechanisms.

9. Hybrid Approach with Non-Autonomic Functions (*)

This section explains how autonomic functions can co-exist with non-autonomic functions, and how a potential overlap is managed. This is all about conflict resolution. Maybe we should re-name that section? [I-D.irtf-nmrg-autonomic-network-definitions] already mentions this. Maybe we don't need much more here. (tbc)

10. Security Considerations

10.1. Threat Analysis

This is a preliminary outline of a threat analysis, to be expanded and made more specific as the various Autonomic Networking specifications evolve.

Since AN will hand over responsibility for network configuration from humans or centrally established management systems to fully distributed devices, the threat environment is also fully distributed. On the one hand, that means there is no single point of failure to act as an attractive target for bad actors. On the other hand, it means that potentially a single misbehaving autonomic device could launch a widespread attack, by misusing the distributed AN mechanisms. For example, a resource exhaustion attack could be launched by a single device requesting large amounts of that resource from all its peers, on behalf of a non-existent traffic load. Alternatively it could simply send false information to its peers, for example by announcing resource exhaustion when this was not the case. If security properties are managed autonomically, a misbehaving device could attempt a distributed attack by requesting all its peers to reduce security protections in some way. In

general, since autonomic devices run without supervision, almost any kind of undesirable management action could in theory be attempted by a misbehaving device.

If it is possible for an unauthorised device to act as an autonomic device, or for a malicious third party to inject messages appearing to come from an autonomic device, all these same risks would apply.

If AN messages can be observed by a third party, they might reveal valuable information about network configuration, security precautions in use, individual users, and their traffic patterns. If encrypted, AN messages might still reveal some information via traffic analysis, but this would be quite limited (for example, this would be highly unlikely to reveal any specific information about user traffic). AN messages are liable to be exposed to third parties on any unprotected Layer 2 link, and to insider attacks even on protected Layer 2 links.

11. IANA Considerations

This document requests no action by IANA.

12. Acknowledgements

Many people have provided feedback and input to this document: Sheng Jiang, Roberta Maglione, Jonathan Hansford.

13. Change log [RFC Editor: Please remove]

00: Initial version.

14. References

[I-D.behringer-anima-autonomic-addressing]
Behringer, M., "An Autonomic IPv6 Addressing Scheme", draft-behringer-anima-autonomic-addressing-00 (work in progress), April 2015.

[I-D.behringer-anima-autonomic-control-plane]
Behringer, M., Bjarnason, S., BL, B., and T. Eckert, "An Autonomic Control Plane", draft-behringer-anima-autonomic-control-plane-02 (work in progress), March 2015.

[I-D.carpenter-anima-gdn-protocol]
Carpenter, B. and B. Liu, "A Generic Discovery and Negotiation Protocol for Autonomic Networking", draft-carpenter-anima-gdn-protocol-03 (work in progress), April 2015.

[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-07 (work in progress), March 2015.

[I-D.jiang-auto-addr-management]
Jiang, S., Carpenter, B., and Q. Qiong, "Autonomic Networking Use Case for Auto Address Management", draft-jiang-auto-addr-management-00 (work in progress), April 2014.

[I-D.pritikin-bootstrapping-keyinfrastructures]
Pritikin, M., Behringer, M., and S. Bjarnason, "Bootstrapping Key Infrastructures", draft-pritikin-bootstrapping-keyinfrastructures-01 (work in progress), September 2014.

Authors' Addresses

Michael H. Behringer (editor)
Cisco

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

Toerless Eckert
Cisco

Email: eckert@cisco.com

Laurent Ciavaglia
Alcatel Lucent
Route de Villejust
Nozay 91620
France

Email: laurent.ciavaglia@alcatel-lucent.com

Bing Liu
Huawei Technologies
Q14, Huawei Campus
No.156 Beiqing Road
Hai-Dian District, Beijing 100095
P.R. China

Email: leo.liubing@huawei.com