

ANIMA WG
Internet-Draft
Intended status: Standards Track
Expires: 5 September 2022

T.T.E. Eckert
Futurewei
M. Boucadair
C. Jacquenet
Orange
M. Behringer
4 March 2022

DNS-SD Compatible Service Discovery in GeneRic Autonomic Signaling
Protocol (GRASP)
draft-eckert-anima-grasp-dnssd-03

Abstract

DNS Service Discovery (DNS-SD) defines a framework for applications to announce and discover services. This includes service names, service instance names, common parameters for selecting a service instance (weight or priority) as well as other service-specific parameters. For the specific case of autonomic networks, GeneRic Autonomic Signaling Protocol (GRASP) intends to be used for service discovery in addition to the setup of basic connectivity. Reinventing advanced service discovery for GRASP with a similar set of features as DNS-SD would result in duplicated work. To avoid that, this document defines how to use GRASP to announce and discover services relying upon DNS-SD features while maintaining the intended simplicity of GRASP. To that aim, the document defines name discovery and schemes for reusable elements in GRASP objectives.

Note to the RFC Editor

Please replace all occurrences of rfcXXXX with the RFC number assigned to this document.

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 <https://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."

Internet-Draft

DNS-SD via GRASP

March 2022

This Internet-Draft will expire on 5 September 2022.

Copyright Notice

Copyright (c) 2022 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 (<https://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 Revised BSD License text as described in Section 4.e of the [Trust Legal Provisions](#) and are provided without warranty as described in the Revised BSD License.

Table of Contents

1.	Overview	3
2.	Terminology	4
3.	Specification	4
3.1.	Service and Name Objectives	4
3.2.	Objective Value Reuseable Elements Structure	5
3.3.	Reuseable Elements	6
3.3.1.	Sender Loop Count	6
3.3.2.	Service Element	6
3.3.3.	Name Element	9
4.	Theory of Operation	11
4.1.	Using GRASP Service Announcements	11
4.2.	Further Comparison with DNS-SD	13
4.3.	Open Issues	13
5.	Security Considerations	14
6.	IANA Considerations	14
7.	Acknowledgements	15
8.	Contributors	15
9.	Change log [RFC Editor: Please remove]	15
9.1.	03 - Refresh	15
9.2.	02 - Revived after charter round 1 finished	15
9.3.	01 -	15
9.4.	00 - Initial version	15
10.	References	15
10.1.	Normative References	15
10.2.	Informative References	16

[1.](#) Overview

DNS Service Discovery (DNS-SD) [[RFC6763](#)] defines a framework for applications to announce and discover services. This includes service names, service instance names, common parameters for selecting a service instance (weight, priority) as well as other service-specific parameters.

GeneRic Autonomic Signaling Protocol (GRASP) [[RFC8990](#)] is intended to also be used for service discovery purposes. Reinventing service discovery for GRASP with a similar set of features would result in duplication of work. Therefore, this document defines how to use GRASP to announce and discover services in a way that inherits DNS-SD features and also tries to be compatible in spirit as much as possible while still maintaining the intended simplicity of GRASP.

The goal of this document is to permit defining service and their parameters once and then use that in GRASP, mDNS and (unicast) DNS. Future work can also define DNS-SD <-> GRASP gateway functions.

This document primarily defines how to perform service discovery across such a GRASP domain leveraging GRASP's options to perform unsolicited flooding of announcements or flooding of requests, and finding the closest service instances. Also, the document allows for automatically discovering DNS-SD servers. Such features is meant to optimize the flooding traffic in some deployments.

The initial use case of this document is to support what in DNS-SD is done via mDNS but in larger networks - GRASP-Domains. Beside the efficient flooding, GRASP provides reliability and security, which are depending on the so called substrate used by GRASP for security and hop-by-hop/end-to-end transport, such as the Autonomic control plane (ACP), [[RFC8994](#)]. Providing compatibility with existing mDNS service announcer or clients is possible, but not described in this version of the document.

The encoding of information chosen in this document does not try to use GRASP solely as a transport layer, but to also leverage the CBOR structure of GRASP messages to natively encode the message elements required for services in a way that is most simple – instead of using GRASP only as, e.g., an encapsulation of otherwise unchanged DNS message encodings. This is done to minimize the amount of coding required (and not require any DNS code unless future gateway functions are required), to increase the simplicity, minimize the amount of data on the wire, and allow easier extensibility. On the downside, the mechanisms provided here do not cover the whole slew of possible options of DNS/DNS-SD, but instead only those deemed to be required. Others can be added later.

In support of service discovery, this document also defines name discovery and schemes for reusable elements in GRASP objectives which are designed to be extensible so that future work that identifies elements required across multiple objectives do not need to define a scheme how to do this.

[2.](#) Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14](#) [[RFC2119](#)] [[RFC8174](#)] when, and only when, they appear in all capitals, as shown here.

This document makes use of terms and concepts defined in [[RFC8990](#)].

[3.](#) Specification

[3.1.](#) Service and Name Objectives

Unsolicited, flooded announcements (M_FLOOD) in GRASP and solicited flooded discovery (M_DISCOVERY) operate on the unit of GRASP technical objectives (identified by 'objective-names' as discussed in [Section 2.10 of \[RFC8990\]](#)). Therefore, a scheme is required to indicate services via 'objective-names'.

Note: Future work may want to reuse the encodings related to services (defined below in this document) inside other (multicast or unicast only) objective exchanges, in which case the service

names are not impacted.

When a technical objective (simply referred to as objective) is meant to be solely about a service name, the objective MUST use an 'objective-name' of 'SRV.<service-name>'. This naming scheme is meant to avoid creating duplicates and, potentially, inconsistent name registrations for those objectives vs. registrations done, for example, for DNS-SD.

When an objective is meant announcement and discovery of a DNS compatible <name> such as "www-internal" in "www-internal.example.com", the objective SHOULD use an objective-name of NAME.<name>. See [Section 3.3.3](#) for more details.

[3.2.](#) Objective Value Reuseable Elements Structure

Because service discovery, as explained in the prior section, needs to utilize different objectives, it requires cross-objective standardized encoding of the elements of services. GRASP does not define standardized message elements for the message body (called "objective-value") of GRASP messages. Therefore, this document introduces such a feature.

```
objective-value  /= { 1*elements }
elements         //= ( @rfcXXXX: { 1*relement } )

relement       = ( relement-codepoint => relement-value )
relement-codepoint = uint
relement-value   = any
```

If an objective relies upon reusable elements, the 'objective-value' MUST be a CBOR map and the reusable elements are found under the key "@rfcXXXX".

Objectives that do not want reusable elements may use any objective-

value format including a CBOR map, but they can not use the "@rfcXXXX" key if they use a map. This approach was chosen as the hopefully least intrusive mechanism given how by nature all of "objective-value" is meant to be defined by individual objective definitions.

The value of "@rfcXXXX" is a map of reusable elements. Each 'relement' has an IANA registered element-name and codepoint (see [Section 6](#)). The element-name is for documentation purposes only, CBOR encodings only use the numeric codepoint for encoding efficiency to minimize the risk for this solution to not be applicable to low-bitrate networks such as in IoT.

Format and semantic of the relement-value is determined by the specification of the reusable element as is the fact whether more than one instances of the same reusable element are permitted.

Reusable elements should be defined to be extensible. The methods used depend on the complexity of the element and the likely need to extend/modify the element with backward or non-backward compatible information. The following is a set of initial options to choose from:

Element values that are a map MUST permit and reserve key value 0 (numerical) for private extensions of the element defined by the individual objective.

Element values that are a map MUST NOT use bareword key values starting with a "_". These too are for private extensions defined by the individual objective.

Element values SHOULD be defined so that additional keys in maps and additional elements at the end of arrays can be ignored by prior versions of the definition. Whenever a newer definition is made for an element where this rule is violated, the element SHOULD be changed in a way for older version recipients to recognize that it is not compatible with it.

One method to indicate compatibility is a traditional version "<major>.<minor>". Within the same <major> version number, increasing <minor> version numbers must be backward compatible. Different <major> version numbers are not expected to be compatible

with each other. If they are, then this can be indicated by including multiple version numbers.

A compressed form of version compatibility information is the use of a simple bitmask element where each bit indicates a version that the represented data is compatible with.

[3.3.](#) Reuseable Elements

[3.3.1.](#) Sender Loop Count

```
relement-codepoint ::= ( &(sender-loop-count:1) => 1..255 )
```

Sender-loop-count is set by the sender of an objective message to the same value as the loop-count of the message. On receipt, distance = (sender-loop-count - loop-count) is the distance of the sender from the receiver in hops. This element can be used for informational purposes in M_FLOOD and M_DISCOVERY messages and may be required to be used in these messages by the specification of other elements (such as the service element described below). This element **MUST** occur at most once. If a receiver expects to use the distance but sender-loop-count was not announced, then distance **SHOULD** be assumed to be 255 by the receiver.

[3.3.2.](#) Service Element

The srv-element (service element) is a reusable element to request or announce a service instance or to request and list service instance names.

```
relement-codepoint ::= ( &(srv-element:2) => context-element )
```

```
context-element = {  
    ?( &(private:0)      => any),  
    ?( &(msg-type:1)     => msg-type),  
    ?( &(service:2)      => tstr),  
    *( &(instance:3)     => tstr),  
    ?( &(domain:4)       => tstr),  
}
```

```

    ?( &(priority:5)      => 0..65535 ),
    ?( &(weight:6)        => 0..65535 ),
    *( &(kvpairs:7)       => { *(tstr: any) },
    ?( &(range:8)         => 0..255 ),
    *( &(clocator:9)      => clocator),
  }
clocator = [ context, locator-option ]
context = cstr
locator-option = ; from GRASP

msg-type = &( describe: 0, describe-request:1,
              enumerate:2, enumerate-request:3 )

```

Service: A service name registered according to [RFC6335](#). If it is not present, then objective-name MUST be SRV.<service-name> where <service-name> is the service-name.

Instance: The <Instance> of a DNS-SD Service Instance Name (<Instance> . <Service> . <Domain>). It is optional, see [Section 4.2](#).

Domain: The equivalent of the <Domain> field of a DNS-SD Service Instance Name. If domain is not present, this is equivalent to ".local" in DNS (as introduced by mDNS) and implies the unnamed "local" domain, which is the GRASP domain across which the message is transmitted.

Priority, Weight: Service Instance selection criteria as defined in [RFC2782](#). If either one is not present, its value defaults to 0.

Kvpairs: Map of key/value pairs that are service parameters in the same format as the key/value pairs in TXT field(s) of DNS-SD TXT records as defined in [RFC6763, section 6.3](#).

Range: Allows to flexibly combine distance and priority/weight based service selection according to the definition of distance in [Section 3.3.1](#).

If min-distance is the distance of the closest service announcer,

and min-range the range announced by it, then the recipient MUST

consider the priority/weight of all service announcers that are not further away than (min-distance + min-range). If not included, range defaults to 255.

If range is announced, the sender-loop-count element MUST also be announced.

Clocator: The "contextual locator" allows to indicate zero or more locators for the indicated service instance. The context element indicates in which context the locator-option is to be resolved. The reserved context value of "" (empty string) indicates the GRASP domain used, aka: the "local" context in which the service announcement is made. The reserved context value of "0" indicates the default routing context of the announcing node. This is often called "global table", "VRF 0" or "default VRF" on nodes using the "VRF" abstraction. Any other value is a string specifying a context such as another VRF.

The mechanism by which originator and recipient of the srv-element agree on common naming for contexts is outside the scope of this specification. The context therefore allows to indicate locators both for the context through which the GRASP message distributed the srv-element (GRASP domain) as well as that for other contexts. Assume the GRASP domain is the ACP, then clocators in ACP would have a context of "", clocators in the global routing table (part of the data-plane) a context of "0", and clocators on other VRFs (also part of data-plane) a clocator that is their string name.

If no locators are indicated, then the locator of the service(s) is the optional locator-option of the GRASP message in which the objective is contained meant to be used for the service(s) indicated and the clocator implied is "".

If locator(s) are indicated, the messages location-option must be ignored for the service (but may be necessary to be present for other purposes of the objective).

Msg-type Type (aka: intention) of the srv-element. If not present, it is assumed to be "describe".

Describe: Describes one service instance. At least one clocator is required for a positive response, all other fields are permitted, but optional. "Describe" is used in M_FLOOD for unsolicited announcements of services (flooded), in M_RESPONSE messages for solicited announcements of a service and in M_NEGOTIATE for negotiated announcements (both unicasted). If clocator is not included, then all fields except service and instance (and msg-

type and private) must not be included and the srv-element provides a negative reply: No information about this service/service instance. This is only permitted in unicasted "describe" messages.

Describe-request: Request for a "describe" reply. It is used in M_DISCOVERY (flooded) for solicited discovery of services or in M_REQ_SYN (unicasted) for negotiated discovery of service instance(s). In "describe-request", only service is mandatory (but can be provided via the objective-name field of the message), and domain is optional. "Instance" is optional. If provided, then the recipient is asked to provide information about the named instance only. All other fields of srv-element are to be ignored by the receiver in this specification, but a semantic for setting them may be introduced in follow-up work, specifically to filter replies by the indicated fields.

"Describe-request" without instance MAY be answered by "Enumerate" (see below) if the responder has so many instances that it thinks the initiator should rather first select one or fewer instances and ask for their description. The sender of the "Describe-request" MUST be prepared to accept that answer and as necessary follow up with "Describe-request" with the instance names of interest.

Enumerate: Used in the same GRASP messages as "describe", but instead of providing information about one service instance, it is listing service instance names. The purpose of enumerate is the same as browsing a service in DNS-SD. It would be followed by some human or automated selection of one or more instances and then a "describe" M_REQ_SYN request for those instances sent to the source of the "enumerate" to learn about the locators and other parameters of the service instances.

In this specification, all fields other than service, instance and domain (and msg-type and private) must be unset in "enumerate".

Enumerate-request: Requests an "enumerate" reply. It is used in the same way as "Describe-request" except that instance would usually not be set (because in that case it is more useful to send a "Describe-request").

3.3.3. Name Element

The NAME,<name> elements is meant to provide basic name resolution comparable to mDNS name resolution for GRASP domains where this is

desirable and no better name resolution exist - for example in the ACP where there is no requirement for DNS.

Because the GRASP service lookup (unlike) DNS does not mandate that nodes have names (not even service instance names), the use of names is primarily meant to support legacy software. New designs should instead look up only services and service instance names, and nodes should announce their names as service instance names for the services they offer:

For example consider a GRASP (ACP) domain of "example.com". The node providing some "www" service could have a name "www-internal" which means GRASP objective NAME.www-internal, that objective value would include primarily the nodes IP address(es) and the port number for the www service would have to be guessed (80). Better, the node would announce GRASP objective SRV.www and the objective value would include the service instance name www-internal and the (TCP) port information (80 or a non-default port).

```
relement-codepoint //= ( &(name-element:3) => context-element )
```

```
context-element //= {  
    *( &name:10)          => tstr),  
}
```

```
ipv6-address-option = [O_IPv4_ADDRESS, ipv6-address]  
ipv4-address-option = [O_IPv6_ADDRESS, ipv6-address]  
locator-option /= ipv4-address-option  
locator-option /= ipv6-address-option
```

Name information is carried in the name-element relement. It is a context-element like the one used for srv-element except that it adds the name component and that it does not permit the service and instance components and that it allows only describe and describe-request values in the msg-type. Clocators MUST use the ipv6-address-option or ipv4-address-option in the locator-option component.

TBD: Unclear if/how we should best formalize the differences in the context element permitted information between services and names. The above is quite informal.

Priority, weight, kvpairs, range (and of course private) MAY be used in describe messages to support multiple instances of the same name, as used for name anycast/prioritycast.

Nodes may have multiple names. These can be listed in the name component. If a nodes names have the notion of a primary name and secondary names then the primary name should be the first in the list of names. In DNS-SD, the name pointed to by CNAME RRs can be considered to be the primary name. A describe-request for a non-primary name SHOULD return in the list of names the requested name and the primary name.

Note that there is no reverse lookup defined in this version of the document (no lookup from IP address to name).

[4.](#) Theory of Operation

[4.1.](#) Using GRASP Service Announcements

TBD: This section contains a range of details that should become normative in later versions.

This section provides a step by step walk-through of how to use GRASP service announcements and compares it to DNS-SD.

The most simple method to use GRASP service discovery is to select (and if still necessary, register) a <service-name> and start one or more agents (e.g.: ASAs) announcing their service instance(s) via GRASP. At minimum, an agent should periodically (default 60 seconds) announce the service instance via GRASP M_FLOOD messages as an objective SRV.<service-name> with a srv-element and a sender-loop-count element (default 255). The ttl of the GRASP message should be 3.5 times the announcement period, e.g.: 210000 msec.

Consumers of the service will use GRASP to learn of the service

instances and select one. This approach is most similar to the use of DNS-SD with mDNS except that the scope of the announcement is a whole GRASP domain (such as the ACP) as opposed to a single IP subnet in mDNS and that mDNS primarily relies on request & reply but in its standard not on periodic unsolicited announcements. We describe here the unsolicited flooding option via M_FLOOD first because it is recommended for services with a dense population of service consumers and it is most simple to describe.

On the service announcer, the parameters priority, weight and range of the service instance can be selected from intent or configuration – or left at default. The default range 255 will result in selection of a random target of the service like in DNS-SD. Setting priority/weight allows to prioritize and weigh the selection as in DNS-SD. Setting range to 0 allows to select the closest target, priority/weight are only compared between targets of the same shortest

distance. Distance based options are not available in DNS-SD because it does not expect that network distance is available to arbitrary DNS-SD client. It is available to GRASP clients though. Using $0 < \text{range} < 255$ allows for a hybrid priority/weight and distance based service selection (e.g.: Select the highest priority instance within a range of 5 hops).

If the service is a non-GRASP service, then the result of the service discovery has to be a transport locator to which the client can open a connection and talk the protocol implied by the service. This transport locator(s) have to be put into the clocator parameter. The context of the clocator would normally be "", aka: the transport locator is in the IP reachability associated with the GRASP domain (e.g.: IPv6 of the ACP for ACP GRASP domain).

If an ACP service is announced via ACP GRASP, then the locator(s) can be O_IPv6_LOCATOR or O_FQDN_LOCATOR. The O_IPv6_LOCATOR is used if the service is defined to be available via some transport layer port (TCP, UDP or other). The determination of the actual transport connection to be used is the same as in DNS-SD: If the transport protocol is not TCP or UDP, it has to be implied by the specification of <service-name> or can be detailed in kvpairs which carries the same information as DNS-TXT TXT RRs of the service. Alternatively, the transport-proto field of the locator can contain any valid IP protocol directly (TBD), which is not possible in DNS-SD.

Like DNS-SD, service discovery via GRASP does not require allocation and use of well-known ports for services. Unlike DNS-SD, there is no need in GRASP to define service instance names or target names. In DNS SD, PTR RRs resolve from a service name to a set of service instance named. SRV and TXT RRs resolve from service instance names to service instance parameters including the target. A target is the DNS host name of the service instance. It gets resolved via A/AAAA RRs to IPv4/IPv6 addresses of the target. In GRASP service discovery, host names are not used. Service instance names are optional too. Service instance names are useful for human diagnostics and human selection of service instances. In fully automated environments, they can be are less important. For diagnostic purposes, it is recommended to give service instances service instance names in GRASP service announcements.

A locator with O_URI_LOCATOR type can be used in GRASP to indicate a URI for the transport method for a service instance. If the URI includes a host part, care must be taken to use only IP addresses in the host part if the context of the GRASP domain does not support host name resolution - such as the ACP - or to use the GRASP name resolution mechanisms described elsewhere in this document. And that the addresses indicated are also reachable in the GRASP domain. For

example, in service announcements across a DULL GRASP domain, only the IPv6 link-local addresses on that subnet must be used (this applies equally when using the O_IPv6_LOCATOR).

Instead of using M_FLOOD to periodically announce service instances, M_DISCOVERY can be used to actively query for service instances. The msg-type type must then be "describe-request". Because no periodic flooding is necessary, this solution is more lightweight for the network when the number of requesting clients is small. Note though that the M_DISCOVERY will terminate as soon as a provider of the objective is found, so the service instances found will be based on distance and therefore selection of instance by priority and weight will not work equally well as with M_FLOOD. Consider for example a central service instance in the NOC that should always be used (for example for centralized operational diagnostics) unless the WAN connection is broken, in which case distributed backup service instances should be used. With the current logic of M_DISCOVERY this is not possible.

[4.2.](#) Further Comparison with DNS-SD

Neither the GRASP SRV.* objective-name, the service name nor any other parameter explicitly indicate the second label "_tcp" or "_udp" of DNS-SD entries. DNS-SD, [RFC6763](#) explains how this is an unnecessary, historic artifact.

This version of the document does not define an equivalent to "_sub" structuring of service enumeration.

This version of the document does not define mechanisms for reverse resolution of arbitrary services: An inquirer may unicast M_SYNC_REC to a node with a series of objectives with specific service names of interest and describe-request, but there is no indication of "ANY" service.

[4.3.](#) Open Issues

TBD: Examine limitations mentioned in "in this version of the text/document".

TBD: The GRASP specification does currently only permit TCP and UDP for the transport-proto element. This draft should expand the GRASP definitions to permit any valid IP protocol. We just need to decide whether this should only apply to the locator in the srv element or also retroactive to the locator-option in GRASP messages (maybe not there ?).

TBD: A fitting CBOR representation for a kvpair key without value needs to be specified so that it can be distinguished from an empty value as outlined in [RFC6763 section 6.4](#).

TBD: In this version, every service/service-instance is an element by itself. Future versions of this document may add more encoding options to allow more compact encoding of recurring fields.

TBD: Is there a way in CDDL to formally define the string names of the relement-codepoint's ?

5. Security Considerations

TBD.

GRASP-related security issues are discussed in [Section 3 of \[RFC8990\]](#).

6. IANA Considerations

This document requests IANA to create a new "GRASP Objective Value Standard Elements" subregistry under the "GeneRIC Autonomic Signaling Protocol (GRASP) Parameters" registry.

The values in this table are names and a unique numerical value assigned to each name. Future values MUST be assigned using the RFC Required policy as dedfined in [Section 4.7 of \[RFC8126\]](#). The numerical value is simply to be assigned sequentially. The following initial values are assigned by this document:

sender-loop-count 1 [defined in rfcXXXX]

srv-element 2 [defined in rfcXXXX]

name-element 3 [defined in rfcXXXX]

This document updates the handling of the "GRASP Objective Names" Table introduced in the GRASP IANA considerations as follows:

Assignments for objective-names of the form "SRV.<text>" and "NAME.<text>" are special.

Assignment of "SRV.<text>" can only be requested if <text> is also a registered service-name according to [RFC6335](#). The specification required for registration of a "GRASP Objective Name" MUST declare that the intended use of the objective name in GRASP is intended to be compatible with the indented use of the registered service name.

Registration of "SRV.<text>" in the "GRASP Objective Name" table is optional, but recommended for all new service-names that are meant to be used with GRASP. Non-registration can for example happen with DNS-SD <-> GRASP gateways that inject pre-existing service-names into

GRASP. Note that according to the GRASP RFC, registration is mandatory, so this exemption for "SRV.<text>" is also an update to that specification.

There MUST NOT be any assignment for objective names of the form "NAME.<text>". These names are simply used by GRASP nodes without registration (just like names in mDNS).

[7.](#) Acknowledgements

[8.](#) Contributors

Brian Carpenter

[9.](#) Change log [RFC Editor: Please remove]

[9.1.](#) 03 - Refresh

[9.2.](#) 02 - Revived after charter round 1 finished

Reviving after ANIMA charter 01 is finished, adding new co-authors, contributors.

Textual improvements, updating references.

[9.3.](#) 01 -

Only refreshing, no changes since -00.

[9.4.](#) 00 - Initial version

[10.](#) References

[10.1.](#) Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", [RFC 6763](#), DOI 10.17487/RFC6763, February 2013, <<https://www.rfc-editor.org/info/rfc6763>>.

- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", [BCP 26](#), [RFC 8126](#), DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in [RFC 2119](#) Key Words", [BCP 14](#), [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8990] Bormann, C., Carpenter, B., Ed., and B. Liu, Ed., "GeneRic Autonomic Signaling Protocol (GRASP)", [RFC 8990](#), DOI 10.17487/RFC8990, May 2021, <<https://www.rfc-editor.org/info/rfc8990>>.

10.2. Informative References

- [RFC8994] Eckert, T., Ed., Behringer, M., Ed., and S. Bjarnason, "An Autonomic Control Plane (ACP)", [RFC 8994](#), DOI 10.17487/RFC8994, May 2021, <<https://www.rfc-editor.org/info/rfc8994>>.

Authors' Addresses

Toerless Eckert
Futurewei Technologies USA Inc.
2220 Central Expressway
Santa Clara, 95050
United States of America
Email: tte+ietf@cs.fau.de

Mohamed Boucadair
Orange
35000 Rennes
France
Email: mohamed.boucadair@orange.com

Christian Jacquenet
Orange
35000 Rennes
France
Email: christian.jacquenet@orange.com

Michael H. Behringer
Email: michael.h.behringer@gmail.com

