intarea
Internet-Draft
Intended status: Informational
Expires: September 14, 2017
B. Bruneau
Ecole polytechnique
P. Pfister
Cisco
D. Schinazi
T. Pauly
Apple
E. Vyncke, Ed.
Cisco
March 13, 2017

Proposals to discover Provisioning Domains
draft-bruneau-intarea-provisioning-domains-00

Abstract

This document describes one possible way for hosts to retrieve additional information about their Internet access configuration. The set of configuration items required to access the Internet is called a Provisioning Domain (PvD) and is identified by a Fully Qualified Domain Name.

This document separates the way of getting the Provisioning Domain identifier, the way of getting the Provisioning Domain information and the potential information contained in the Provisioning Domain.

Copyright Notice

Table of Contents

1. Introduction

It has become very common in modern networks that hosts have Internet or more specific access through different networking interfaces, tunnels, or next-hop routers. The concept of Provisioning Domain (PvD) was defined in RFC7556 [RFC7556] as a set of network configuration information which can be used by hosts in order to access the network. In this document, PvDs are associated with a Fully Qualified Domain Name (called PvD ID) which is used within the host to identify correlated sets of configuration data and also used to retrieve additional information about the services that the network provides.

Devices connected to the Internet through multiple interfaces would typically be provisioned with one PvD per interface, but it is worth noting that multiple PvDs with different PvD IDs could be provisioned on any host interface, as well as noting that the same PvD ID could be used on different interfaces in order to inform the host that both PvDs, on different interfaces, ultimately provide equivalent services.

This document proposes multiple methods allowing the host to to retrieve the PvD ID associated with a set of networking discover the PvD and retrieve the PvD information. It also explains configuration as well as the methods and format in order to retrieve some of the parameters that can describe a PvD.

2. Terminology

PvD A provisioning domain, usually with a set of provisioning domain information; for more information, see [RFC7556].

2.1. Requirements Language

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 RFC 2119 [RFC2119].

3. Retrieving the PvD ID

In this document, each provisioning domain is identified by a PvD ID. The PvD ID is a Fully Qualified Domain Name which belongs to the network operator to avoid conflicts among network operators. The same PvD ID can exist in several access networks if the set of configuration information is identical in all those networks (such as in all home networks of a residential subscriber). Within a host, the PvD ID SHOULD be associated to all the configuration information associated to this PvD ID; this allows for easy update and removal of information while keeping a consistent state.

This section assumes that IPv6 Router Advertisements are used to discover the PvD ID and explains why this technique was selected.

3.1. Using One Router Advertisement per PvD

Hosts receive implicit PvDs by the means of Router Advertisements (RA).

A router MAY add a single PvD ID Option in its RAs. The PvD ID specified in this option is then associated with all the Prefix Information Options (PIO) included in the RA (albeit it is expected that only one PIO will be included in the RA). All other information contained in the RA (notably the RDNSS and Route Information Option) are to be associated with the PvD ID. The set of information contained in the RA forms the bootstrap (or hint) PvD. A new RA option will be required to convey the PvD ID.

When a host receives an RA which does not include a PvD ID Option, the set of information included in the RA (such as Recursive DNS server, IPv6 prefix) is attached to an implicit PvD identified by the local interface ID on which the RA is received, and by the link-local address of the router sending the RA.

In the cases where a router should provide multiple independent PvDs to all hosts, including non-PvD aware hosts, it should send multiple RAs, as proposed in [I-D.bowbakova-rtgwg-enterprise-pa-multihoming] using different source link-local addresses (LLA); the datalink layer (MAC) address could be the same for all the different RA. If the router is actually a VRRP instance, then the procedure is identical

except that the virtual link-layer address is used as well as virtual link-layer addresses.

Using RA allows for an early discovery of the PvD ID as it is early in the interface start-up. As RA is usually processed in the kernel, this requires a host OS upgrade. The RA SHOULD contain other PvD information as explained in section Section 4.1.

3.2. Rationale for not selecting other techniques

There are other techniques to discover the PvD ID that were not selected by the authors and reviewers, this section explains why. The design goal was to be as reliable as possible (do not depend on Internet connectivity) and as fast as possible.

3.2.1. Using DNS-SD

For each received RA including a RDNSS option as well as a DNS search list option, the host MAY retrieve the PvD ID by querying the configured DNS server for records of type PTR associated with _pvd.<DNS search name>. If a PvD ID is configured, the DNS recursive resolver MUST reply with the PvD ID as a PTR record. NXDOMAIN is returned otherwise.

When the RDNSS address is link-local, the host MAY retrieve the PvD ID before configuring its global scope address(es).

Relying on a valid DNS service at the interface bootstrap can lead into delay to start the interface or starting without enough information: for example when the RDNSS is a non local address and there is no Internet connectivity.

3.2.2. Using Reverse DNS lookup

[I-D.stenberg-mif-mpvd-dns] proposes a solution to get the name of the PvD using a reverse DNS lookup based on the host global address(es). It merely relies on prepending a well-known prefix '_pvd' to the reverse lookup, for example ' _pvd....ip6.arpa.'.

However, the PvD information is typically provided by the network operator, whereas the reverse DNS zone could be delegated from the operator to the network user, in which case it would not work.

It also requires a fully functional global address to retrieve the information which may be too late for a correct host configuration. One advantage is that it does not require any change in the IPv6 protocol and no change in the host kernel or even in the CPE.

3.3. IoT Considerations

TBD: should state that when end-host (IoT) cannot impletement completely this RFC it MAY select any of the PvD or the router SHOULD send a single unicast RA (hence a single PvD) in response to the RS or none if it detects that it cannot offer the right set of network services.

3.4. Linking IPv4 Information to an IPv6 PvD

The document describes IPv6-only PvD but there are multiple ways to link the set of IPv4 configuration information received by DHCPv4:

- correlation based on the data-link layer address of the source, if the IPv6 RA and the DHCPv4 response have the same data-link layer address, then the information contained in the IPv4 DHCP can be linked to the IPv6 PvD;

- correlation based on the interface when there is no data-link address on the link (such as a 3GPP link), then the information contained in the IPv4 PDP context can be linked to the IPv6 PvD (*** TO BE VERIFIED before going -01);

- correlation based on the DNS search list, if the DNS search lists are identical between the IPv6 RDNSS and the DHCPV4 response, then the information contained in the IPv4 DHCP response can be linked to the IPv6 PvD.

The correlation could be useful for some PvD information such as Internet reachability, use of captive portal, display name of the PvD, ...

In cases where the IPv4 configuration information could not be associated with a PvD, hosts MUST consider it as attached to an independent implicit PvD containing no other information than what is provided through DHCPv4.

4. Getting the full set of PvD information

Once the PvD ID is known, it MAY be used to retrieve additional information. PvD Information is modeled as a key-value dictionary which keys are ASCII strings of arbitrary length, and values are either strings (encoding can vary), ordered list of values (recursively), or a dictionary (recursively).

The PvD Information may be retrieved from multiple sources (from the bootstrap PvD contained in the RA to the secondary/extended PvD described in this section); the PvD ID is then used to correlate the

information from different sources. The way a host should operate when receiving conflicting information is TBD but it SHOULD at least override information from less authenticated sources (RA) by more authenticated sources (via TLS).

4.1. Using the PvD Bootstrap Information Option

Routers MAY transmit, in addition to the PvD ID option, a PvD Bootstrap Information option, containing a first subset of PvD information. The additional pieces of bootstrap PvD information data set are transmitted using the short-hand notation proposed in Section 5. This requires another RA option.

As there is a size limit on the amount of information a single RA can convey, it is likely that the PvD Bootstrap Information option may not contain the whole set of PvD Information. The set of PvD information included in the RA is called PvD Bootstrap Information.

4.2. Downloading a JSON file over HTTPS

The host SHOULD try to download a JSON formatted file over HTTPS in order to get more PvD information.

The host MUST perform an HTTP query to https://<PvD-ID>/v1.json. If the HTTP status of the answer is greater than 400 the host MUST abandon and consider that there is no additional PvD information. If the HTTP status of the answer is between 300 and 400 it MUST follow the redirection(s). If the HTTP status of the answer is between 200 and 300 the host MAY get a file containing a single JSON object.

The host MUST respect the cache information in the HTTP header, if any, and at expiration of the downloaded object, it must fetch a fresher version if any.

4.2.1. Advantages

The JSON format allows advanced structures.

It can be secured using HTTPS (and DNSSEC).

It is easier to update a file on a web server than to edit DNS records. It can be especially important if we want providers to be able to often update the remaining phone plan of the user.

4.2.2. Disadvantages

It is slower than using DNS because HTTPS uses TCP and TLS and needs more packets to be exchanged to get the file.

An additional HTTPS server must be deployed and configured.

4.3. Using DNS TXT ressource records (not selected)

This approach was not selected during the design team meeting but has kept here for reference, it will be removed after global consensus is reached.

The host could perform a DNS query for TXT resource records (RR) for the FQDN used as PvD ID (alternatively for _pvd.<PvD-ID>). For each retrieved PvD ID, the DNS query MUST be sent to the DNS server configured from the same router advertisement as the PvD ID. Syntax of the TXT response is defined in Section 5 (Section 5).

4.3.1. Advantages

It requires a single round-time trip in order to retrieve the PvD Information.

It can be secured using DNSSEC.

4.3.2. Disadvantages

A TXT record is limited to 65535 characters in theory but large size of TXT records could require either DNS over TCP (so loosing the 1-RTT advantage) or fragmented UDP packets (which could be dropped by a bad choice of security policy). Large TXT records could also be used to mount an amplification attack.

4.3.3. Using DNS SRV ressource records

It is expected that the DNS TXT records will be sufficient for the host to configure itself with basic networking and policy configuration. Nevertheless, if further information is required, or when a different security model shall be used to access the PvD Information, a SRV Resource Record including a full URL MAY be included as a response, expecting the host to query this URL in order to retrieve additional PvD information.

5. PvD Information

PvD information is a set of key-value pairs. Keys are ASCII character strings. Values are either a character string, an ordered list of values, or an embedded dictionary. Value types and default behavior with respect to some specific keys MAY be further specified (recursively). Some keys have a default value as described in the following sections. When there is an expiration time in a PvD, then the information MUST be refreshed before the expiration time. The behavior of a host when the refresh operation is not successful is TBD.

Nodes using the PvD MUST support the two encodings:

JSON syntax for the complete set of PvD information;

short-hand notation for the bootstrap PvD.

When the PvD information is transferred as a JSON file, then the key used is the second column of the following table. The syntax of the JSON file is obvioulsy JSON and is richer than the short-hand notation specified in the next paragraph.

When transmitting more information than the PvD ID in the RA (or when DNS TXT resource records are used), the shorthand notataion for PvD information is used and consists of a string containing several "key=value;" substrings. The "key" is the first column of the following tables, the value is encoded as:

Shorthand notation for values:

integer: expressed in decimal format with a '.' (dot) used for decimals;

string: expressed as UTF-8 encoded string, delimited by single quote character, the single quote character can be expressed by two consecutive single quote character;

boolean: expressed as '0' for false and '1' for true;

IPv6 address: printed as RFC5952 [RFC5952].

5.1. PvD Name

PvD SHOULD have a human readable name in order to be presented on a GUI. The name can also be localized.

DNS TXT key/Bootstrap PvD key	JSON key	Description	Type	JSON Example
n	name	User-visible service name, SHOULD be part of the bootstrap PvD	human-readable UTF-8 string	"Foobar Service"
nl10n	localizedName	Localized user-visible service name, language can be selected based on the HTTP Accept-Language header in the request.	human-readable UTF-8 string	"Service Blabla"

5.2. Trust of the bootstrap PvD

The content of the bootstrap PvD (from the original RA) cannot be trusted as it is not authenticated. But, the extended PvD can be associated with the PvD ID (as the PvD ID is used to construct the extended PvD URL) and trusted by the used of TLS. The extended PvD SHOULD therefore include the following information elements and, if they are present, the host MUST verify that the all PIO of the RA fits into the master prefix list. If any PIO prefix from the bootstrap PvD does not fit in the master prefix array, then all information received by the bootstrap PvD must be invalidated. In short, the masterIPv6Prefix received over TLS is used to authenticate the bootstrap PvD.

The values of the bootstrap PvD (RDNSS, ...) are overwritten by the values contained in the trusted extended PvD if they are present.

DNS TXT key	JSON key	Description	Type	JSON Example
mp6	masterIpv6Prefix	All the IPv6 prefixes linked to this PvD (such as a /29 for the ISP).	Array of IPv6 prefixes	["2001:db8::/32"]

5.3. Reachability

The following set of keys can be used to specify the set of services for which the respective PvD should be used. If present they MUST be honored by the client, i.e., if the PvD is marked as not usable for Internet access (walled garden), then it MUST NOT be used for Internet access. If the usability is limited to a certain set of domain or address prefixes (typical VPN access), then a different PvD MUST be used for other destinations.

DNS TXT key	JSON key	Description	Type	JSON Example
s	noInternet	Internet inaccessible	boolean	true
cp	captivePortal	Presence of a captive portal	boolean	false
z	dnsZones	DNS zones accessible and searchable	array of DNS zone	["foo.com","sub.bar.com"]
6	prefixes6	IPv6-prefixes accessible via this PvD	array of IPv6 prefixes	["2001:db8:a::/48","2001:db8:b:c::/64"]
4	prefixes4	IPv4-prefixes accessible	array of IPv4 prefixes in CIDR reachable via this PvD	["192.0.2.0/24","2.3.0.0/16"]

5.4. DNS Configuration

The following set of keys can be used to specify the DNS configuration for the respective PvD. If present, they MUST be honored and used by the client whenever it wishes to access a resource described by the PvD.

DNS TXT key	JSON key	Description	Value	JSON Example
r	dnsServers	Recursive DNS server	array of IPv6 and IPv4 addresses	["2001:db8::1","192.0.2.2"]
d	dnsSearch	DNS search domains	array of search domains	["foo.com","sub.bar.com"]

5.5. Connectivity Characteristics

NOTE: open question to the authors/reviewers: should this document include this section or is it useless?

The following set of keys can be used to signal certain characteristics of the connection towards the PvD.

They should reflect characteristics of the overall access technology which is not limited to the link the host is connected to, but rather a combination of the link technology, CPE upstream connectivity, and further quality of service considerations.

DNS TXT key	JSON key	Description	Type	JSON Example
tp	throughputMax	Maximum achievable throughput (e.g. CPE downlink/uplink)	object({down(int), up(int)}) in kb/s	{"down": 10000, "up": 5000}
lt	latencyMin	Minimum achievable latency	object({down(int), up(int)}) in ms	{"down": 10, "up": 20}
rl	reliabilityMax	Maximum achievable reliability	object({down(int), up(int)}) in 1/1000	{"down": 1000, "up": 800}
cp	captivePortal	Captive portal	URL of the portal	"https://example.com"
nat	NAT	IPv4 NAT in place	boolean	true
natto	NAT Time-out	The value in seconds of the NAT time-out	Integer	30
srh	segmentRoutingHeader	The IPv6 Segment Routing Header to be used between the IPv6 header and	Binary string	...

		any other headers when using this PvD		
srhDNS	segmentRoutingHeaderDnsFQDN	The DNS FQDN which is used to retrieved the actual IPv6 Segment Routing Header to be used between the IPv6 header and any other headers when using this PvD	Ascii string	srh.pvd-foo.example.org
cost	cost	Cost of using the connection	object	See Section 5.6

5.6. Connection monetary cost

NOTE: This section is included as a request for comment on the potential use and syntax.

The billing of a connection can be done in a lot of different ways. The user can have a global traffic threshold per month, after which his throughput is limited, or after which he/she pays each megabyte. He/she can also have an unlimited access to some websites, or an unlimited access during the weekends.

We propose to split the final billing in elementary billings, which have conditions (a start date, an end date, a destination IP address...). The global billing is an ordered list of elementary billings. To know the cost of a transmission, the host goes through the list, and the first elementary billing whose the conditions are fulfilled gives the cost. If no elementary billing conditions match the request, the host MUST make no assumption about the cost.

5.6.1. Conditions

Here are the potential conditions for an elementary billing. All conditions MUST be fulfill.

Note: the final version should use short-hand key names.

Key	Description	Type	JSON Example
beginDate	Date before which the billing is not valid	ISO 8601	"1977-04-22T06:00:00Z"
endDate	Date after which the billing is not valid	ISO 8601	"1977-04-22T06:00:00Z"
domains	FQDNs whose the billing is limited	array(string)	["deezer.com","spotify.com"]
prefixes4	IPv4 prefixes whose the billing is limited	array(string)	["78.40.123.182/32","78.40.123.183/32"]
prefixes6	IPv6 prefixes whose the billing is limited	array(string)	["2a00:1450:4007:80e::200e/64"]

5.6.2. Price

Here are the different possibilities for the cost of an elementary billing. A missing key means "all/unlimited/unrestricted". If the elementary billing selected has a trafficRemaining of 0 kb, then it means that the user has no access to the network. Actually, if the last elementary billing has a trafficRemaining parameter, it means that when the user will reach the threshold, he/she will not have access to the network anymore.

Key	Description	Type	JSON Example
pricePerGb	The price per Gigabit	float (currency per Gb)	2
currency	The currency used	ISO 4217	"EUR"
throughputMax	The maximum achievable throughput	float (kb/s)	1000
trafficRemaining	The traffic remaining	float (kb)	96000000

5.6.3. Examples

Example for a user with 20 GB per month for 40 EUR, then reach a threshold, and with unlimited data during weekends and to deezer:


```
[
  {
    "domains": ["deezer.com"]
  },
  {
    "prefixes4": ["78.40.123.182/32","78.40.123.183/32"]
  },
  {
    "beginDate": "2016-07-16T00:00:00Z",
    "endDate": "2016-07-17T23:59:59Z",
  },
  {
    "beginDate": "2016-06-20T00:00:00Z",
    "endDate": "2016-07-19T23:59:59Z",
    "trafficRemaining": 96000000
  },
  {
    "throughputMax": 1000
  }
]
```


If the host tries to download data from deezer.com, the conditions of the first elementary billing are fulfilled, so the host takes this elementary billing, finds no cost indication in it and so deduces that it is totally free. If the host tries to exchange data with youtube.com and the date is 2016-07-14T19:00:00Z, the conditions of the first, second and third elementary billing are not fulfilled. But the conditions of the fourth are. So the host takes this

elementary billing and sees that there is a threshold, 12 GB are remaining.

Another example for a user abroad, who has 3 GB per year abroad, and then pay each MB:

```
[
  {
    "beginDate": "2016-02-10T00:00:00Z",
    "endDate": "2017-02-09T23:59:59Z",
    "trafficRemaining": 9200000
  },
  {
    "pricePerGb": 30,
    "currency": "EUR"
  }
]
```

5.7. Private Extensions

keys starting with "x-" are reserved for private use and can be utilized to provide vendor-, user- or enterprise-specific information. It is RECOMMENDED to use one of the patterns "x-FQDN-KEY" or "x-PEN-KEY" where FQDN is a fully qualified domain name or PEN is a private enterprise number [PEN] under control of the author of the extension to avoid collisions.

5.8. Examples

5.8.1. Using JSON

```
{
  "name": "Orange France",
  "localizedName": "Orange France",
  "dnsServers": ["8.8.8.8", "8.8.4.4"],
  "throughputMax": {
    "down": 100000,
    "up": 20000
  },
  "cost": [
    {
      "domains": ["deezer.com"]
    },
    {
      "prefixes4": ["78.40.123.182/32","78.40.123.183/32"]
    },
    {
      "beginDate": "2016-07-16T00:00:00Z",
      "endDate": "2016-07-17T23:59:59Z",
    },
    {
      "beginDate": "2016-06-20T00:00:00Z",
      "endDate": "2016-07-19T23:59:59Z",
      "trafficRemaining": 96000000
    },
    {
      "throughputMax": 1000
    }
  ]
}
```


5.8.2. Using DNS TXT records


```
n=Orange France
r=8.8.8.8,8.8.4.4
tp=100000,20000
cost+0+domains=deezer.com
cost+1+prefixes4=78.40.123.182/32,78.40.123.183/32
cost+2+beginDate=2016-07-16T00:00:00Z
cost+2+endDate=2016-07-17T23:59:59Z
cost+3+beginDate=2016-06-20T00:00:00Z
cost+3+endDate=2016-07-19T23:59:59Z
cost+3+trafficRemaining=96000000
cost+4+throughputMax=1000
```

6. Use case examples

TBD: 1 or 2 examples when PvD are critical

6.1. Multihoming

First example could be multihoming (very much in-line with bowbakova draft).

6.2. VPN/Extranet example

using PvD to reach a specific destination (such as VPN or extranet).

7. Security Considerations

While the PvD ID can be forged easily, if the host retrieve the extended PvD via TLS, then the host can trust the content of the extended PvD and verifies that the RA prefix(es) are indeed included in the master prefixed of the extended PvD.

8. Acknowledgements

Many thanks to M. Stenberg and S. Barth: Section 5.3, Section 5.5 and Section 5.7 are from their document [I-D.stenberg-mif-mpvd-dns].

Thanks also to Ray Bellis, Lorenzo Colitti, Marcus Keane, Erik Kline, Jen Lenkova, Mark Townsley and James Woodyatt for useful and interesting brainstorming sessions.

9. References

9.1. 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, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <http://www.rfc-editor.org/info/rfc5952>.

[RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <http://www.rfc-editor.org/info/rfc7556>.

9.2. Informative references

[I-D.bowbakova-rtgwg-enterprise-pa-multihoming]
Baker, F., Bowers, C., and J. Linkova, "Enterprise Multihoming using Provider-Assigned Addresses without Network Prefix Translation: Requirements and Solution", draft-bowbakova-rtgwg-enterprise-pa-multihoming-01 (work in progress), October 2016.

[I-D.stenberg-mif-mpvd-dns]
Stenberg, M. and S. Barth, "Multiple Provisioning Domains using Domain Name System", draft-stenberg-mif-mpvd-dns-00 (work in progress), October 2015.

[PEN] IANA, "Private Enterprise Numbers", <https://www.iana.org/assignments/enterprise-numbers>.

Authors' Addresses

Basile Bruneau
Ecole polytechnique
Vannes 56000
France

Email: basile.bruneau@polytechnique.edu

Pierre Pfister
Cisco
11 Rue Camille Desmoulins
Issy-les-Moulineaux 92130
France

Email: ppfister@cisco.com

David Schinazi
Apple

Email: dschinazi@apple.com

Tommy Pauly
Apple

Email: tpauly@apple.com

Eric Vyncke (editor)
Cisco
De Kleetlaan, 6
Diegem 1831
Belgium

Email: evyncke@cisco.com