Dynamic Host Configuration (DHC)
Internet-Draft
Intended status: Informational
Expires: January 20, 2014

T. Mrugalski
ISC
K. Kinnear
Cisco

July 19, 2013

DHCPv6 Failover Requirements
draft-ietf-dhc-dhcpv6-failover-requirements-07

Abstract

The DHCPv6 protocol, defined in [RFC3315] allows for multiple servers to operate on a single network, however it does not define any way the servers could share information about currently active clients and their leases. Some sites are interested in running multiple servers in such a way as to provide increased availability in case of server failure. In order for this to work reliably, the cooperating primary and secondary servers must maintain a consistent database of the lease information. [RFC3315] allows for but does not define any redundancy or failover mechanisms. This document outlines requirements for DHCPv6 failover, enumerates related problems, and discusses the proposed scope of work to be conducted. This document does not define a DHCPv6 failover protocol.

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 January 20, 2014.

Table of Contents

1. Introduction

The DHCPv6 protocol, defined in [RFC3315] allows for multiple servers to be operating on a single network, however it does not define how the servers can share the same address and prefix delegation pools and allow a client to seamlessly extend its existing leases when the original server is down. [RFC3315] provides for these capabilities, but does not document how the servers cooperate and communicate to provide this capability. Some sites are interested in running multiple servers in such a way as to provide redundancy in case of server failure. In order for this to work reliably, the cooperating primary and secondary servers must maintain a consistent database of the lease information.

This document discusses failover implementations scenarios, failure modes, and synchronization approaches to provide background to the list of requirements for a DHCPv6 failover protocol. It then defines a minimum set of requirements that failover must provide to be useful, while acknowledging that additional features may be specified as extensions. This document does not define a DHCPv6 failover protocol.

The failover model, to which these requirements apply, will initially be a pairwise "hot standby" model (see Section 4.1) with a primary server used in normal operation switching over to a backup secondary server in the event of failure. Optionally, a secondary server may provide failover service for multiple primary servers. However the requirements will not preclude a future load-balancing extension where there is a symmetric failover relationship.

The DHCPv6 failover concept borrows heavily from its DHCPv4 counterpart [dhcpv4-failover] that never completed standardization process, but has several successful, operationally proven vendor-specific implementations. For a dicussion about commonalities and differences, see Section 6.

2. Definitions

This section defines terms that are relevant to DHCPv6 failover.

Definitions from [RFC3315] are included by reference. In particular, client means any device e.g., end user host, CPE (Customer Premises Equipment) or other router that implements client functionality of the DHCPv6 protocol. A server means a DHCPv6 server, unless explicitly noted otherwise. A relay is a DHCPv6 relay.

A binding (or client binding) is a group of server data records containing the information the server has about the addresses in an IA (Identity Assocation, see Section 10 of [RFC3315]) or configuration information explicitly assigned to the client. Configuration information that has been returned to a client through a policy - for example, the information returned to all clients on the same link - does not require a binding.

DDNS - an abbreviation for "Dynamic DNS", which refers to the capability to update a DNS server's name database using the on-the-wire protocol defined in [RFC2136]. Clients and servers can negotiate the scope of such updates as defined in [RFC4704].

Failover - an ability of one partner to continue offering services provided by another partner, with minimal or no impact on clients.

FQDN - a fully qualified domain name. A fully qualified domain name generally is a host name with at least one domain label under the top-level domain. For example "dhcp.example.org" is a fully qualified domain name.

High Availability - a desired property of DHCPv6 servers to continue providing services despite experiencing unwanted events such as server crashes, link failures, or network partitions.

Load Balancing - the ability for two or more servers to each process some portion of the client request traffic in a conflict-free fashion.

Lease - an IPv6 address, an IPv6 prefix or other resource that was assigned ("leased") by a server to a specific client. A lease may include additional information, like associated fully qualified domain name (FQDN) and/or information about associated DNS updates. A client obtains a lease for a specified period of time (valid lifetime).

Partner - A "partner", for the purpose of this document, refers to a failover server, typically the other failover server in a failover relationship.

Stable Storage - each DHCP server is required to keep its lease database in some form of storage (known as "stable storage") that will be consistent throughout reboots, crashes and power failures.

Partner Failure - A power outage, unexpected shutdown, crash or other type of failure that renders a partner unable to continue its operation.

3. Scope of work

In order to fit within the IETF process effectively and efficiently, the standardization effort for DHCPv6 failover is expected to proceed with the creation of documents of increasing specificity. It begins with this document specifying the requirements for DHCPv6 failover ("requirements document"). Later documents are expected to address the design of the DHCPv6 failover protocol ("design document"), and if sufficient interest exists, the protocol details required to implement the DHCPv6 failover protocol itself ("protocol document"). The goal of this partitioning is, in part, to ease the validation, review, and approval of the DHCPv6 failover protocol by presenting it in comprehensible parts to the larger community.

Additional documents describing extensions may also be defined.

DHCPv6 Failover requirements are presented in Section 7.

3.1. Alternatives to Failover

There are many scenarios when it seems that a failover capability would be useful. However, there are often much simpler approaches that will meet the required goals. This section documents examples where failover is not really needed.

3.1.1. Short-lived addresses

There are cases when IPv6 addresses are used only for a short time, but there is a need to have high degree of confidence that those addresses will be served. A notable example is PXE: Pre eXecution Environment [RFC5970]. This is a mechanism for obtaining configuration early in the process of bootstrapping over the network.

The PXE BIOS acquires an address in order to load the operating system image and continue booting. Address and possibly other configuration parameters are used during the boot process and are discarded thereafter. Any lack of available DHCPv6 service at this time will prevent such devices from booting.

Instead of deploying failover, it is better to use the much simpler preference mechanism, defined in [RFC3315]. For example, consider two or more servers with each having a distinct preference set (e.g., 10 and 20). Both will answer to a client's request. The client should choose the one with larger preference value. In case of failure of the most preferred server, the next server will keep responding to clients' queries. This approach is simple to deploy, but does not offer lease stability, i.e., in case of server failure, clients' addresses and prefixes will change.

3.1.2. Redundant servers

In some cases the desire to deploy failover is motivated by high availability, i.e., to continue providing services despite server failure. If there are no additional requirements, that goal may be fulfilled with simply deploying two or more independent servers on the same link.

There are several well-documented approaches showing how such a deployment could work. They are discussed in detail in [RFC6853]. Each of those approaches is simpler to deploy and maintain than full failover.

3.1.3. Distributed databases

Some servers may allow their lease database to be stored in external databases. Another possible alternative to failover is to configure two servers to connect to the same distributed database.

Care should be taken to understand how inconsistencies are solved in such database backends and how such conflict resolutions affect DHCPv6 server operation.

It is also essential to use only a database that provides equivalent reliability and failover capability. Otherwise the single point of failure is only moved to a different location (database rather than DHCPv6 server). Such a configuration does not improve redundancy, but significantly complicates deployment.

A common miscoception regarding database-based redundancy is the assumption that a conflict resolution after recovering from a network partition is not necessary. To explain that fallacy, let's consider an example where there is a very small pool with only one address. There are two servers, each connected to a co-located database node (i.e., running on the same hardware). Network partition occurs. Each server is operating, but has lost connection to its partner. Two clients request an address, one from each server. Each server consults its database and discovers that only one address is available, so it is assigned to the client. Unfortunately, each server assigned the same address to a different client. Making the scenario more realistic (millions of addresses rather than one) just decreased failure probability, but did not eliminate the underlying issue.

Any solution that involves a distributed database implementation of DHCPv6 failover must take into account the requirements for security. See Section 8 for additional information.

3.1.4. Load Balancing

Sometimes the desire to deploy more than one server is based on the assumption that they will share the client traffic. Administrators that are interested in such a capability are advised to deploy a load balancing mechanism, defined in [I-D.ietf-dhc-dhcpv6-load-balancing].

4. Failover Scenarios

The following section provides several examples of deployment scenarios and use cases that may be associated with capabilities commonly referred to as failover. These scenarios may be in or out of scope for the DHCPv6 failover protocol to which this document's requirements apply; they are enumerated here to provide a common basis for discussion.

4.1. Hot Standby Model

In the simplest case, there are two partners that are connected to the same network. Only one of the partners ("primary") provides services to clients. In case of its failure, the second partner ("secondary") continues handling services previously handled by first partner. As both servers are connected to the same network, a partner that fails to communicate with its partner while also receiving requests from clients may assume with high probability that its partner is down and the network is functional. This assumption may affect its operation.

4.2. Geographically Distributed Failover

Servers may be physically located in separate locations. A common example of such a topology is where a service provider has at least a regional high performance network between geographically distributed datacenters. In such a scenario, one server is located in one datacenter and its failover partner is located in another remote datacenter. In this scenario, when one partner finds that it cannot communicate with the other partner, it does not necessarily mean that the other partner is down.

4.3. Load balancing

A desire to have more than one server in a network may also be created by the desire to have incoming traffic be handled by several servers. This decreases the load each server must endure when all servers are operational. Although such a capability does not, strictly, require failover - it is clear that failover makes such an architecture more straightforward.

Note that in a load balancing situation which includes failover, each individual server must be able to handle the full load normally handled by both servers working together, or there is not a true increase in availability.

4.4. 1-to-1, m-to-1 and m-to-n models

A failover relationship for a specific network is provided by two failover partners. Those partners communicate with each other and back up all pools. This scenario is sometimes referred to as the 1-to-1 model and is considered relatively simple. In larger networks one server may be participating in several failover relationships, i.e., it provides failover for several address or prefix pools, each served by separate partners. Such a scenario can be referred to as m-to-1. The most complex scenario - m-to-n - assumes that each partner participates in multiple failover relationships.

4.5. Split prefixes

Due to the extensive IPv6 address space, it is possible to provide semi-redundant service by splitting the available pool of addressees into two or more non-overlapping pools, with each server handling its own smaller pool. Several versions of such a scenario are discussed in [RFC6853].

4.6. Long lived connections

Certain nodes may maintain long lived connections. Since the IPv6 address space is large, techniques exist (e.g., [RFC6853]) that use the easy availability of IPv6 addresses in order to provide increased DHCPv6 availability. However, these approaches do not generally provide for stable IPv6 addresses for DHCPv6 clients should the server with which the client is interacting become unavailable.

The obvious benefit of stable addresses is the ability to update DNS infrequently. While the DNS can be updated every time an IPv6 address changes, it introduces delays and (depending on DNS configuration) old entries may be cached for prolonged periods of time.

The other benefit of having a stable address is that many monitoring solutions provide statistics on a per IP basis, so IP changes make measuring characteristics of a given box more difficult.

4.7. Partial server communication loss

There is a scenario where the DHCPv6 server may be configured to serve clients on one network adapter and communicate with a partner

server (server to server traffic) on a different network adapter. In this scenario, if the server loses connectivity on the network adapter used to communicate with the clients because of network adapter (hardware) failure, there is no intimation of the loss of service to the partner in the DHCPv6 failover protocol. Since the servers are able to communicate with each other, the partner remains ignorant of the loss of service to clients.

5. Principles of DHCPv6 Failover

This section describes important issues that will affect any DHCPv6 failover protocol. This section is not intended to define implementation details, but rather high level concepts and issues that are important to DHCPv6 failover. These issues form a basis for later documents which deal with the solutions to these issues.

The general failover concept assumes that there are backup servers that can provide service in case of a primary server failure. In theory there could be more than one backup server that could take up the role if such a need arise. However, having more than two servers introduces a very difficult issue of synchronizing between partners. In the case of just a pair of cooperating servers, the notification and processes can result in only one of two states: fully successful (got response from a partner) and total failure (no response, failure event occurred). Were there more than two partners participating in a relationship, there would be intermediate, inconsistent states where some partners had updated their state and some had not. This would greatly complicate protocol design, and would give little advantage in return. Therefore an approach that assumes a pair of cooperating servers was chosen.

5.1. Failure modes

This section documents failure modes.

5.1.1. Server Failure

Servers may become unresponsive due to a software crash, hardware failure, power outage or any number of other reasons. The failover partner will detect such event due to lack of responses from the down partner. In this failure mode, the assumption is that the server is the only equipment that is off-line and all other network equipment is operating normally. In particular, communication between other nodes is not interrupted.

When working under the assumption that this is the only type of failure that can happen, the server may safely assume that its

partner unreachability means that it is down, so other nodes (clients in particular) are not able to reach it either and no services are provided.

It should be noted that recovery after the failed server is brought back on-line is straightforward, due to the fact that it just needs to download current information from the lease database of the healthy partner and there is no conflict resolution required.

This is by far the most common failure mode between two failover partners.

When the two servers are located physically close to each other, possibly in the same room, the probability that a failure to communicate between failover partners is due to server failure is increased.

5.1.2. Network partition

Another possible cause of partner unreachability is a failure in the network that connects the two servers. This may be caused by failure of any kind of network equipment: router, switch, physical cables, or optic fibers. As a result of such a failure the network is split into two or more disjoint sections (partitions) that are not able to communicate with each other. Such an event is called network partition. If failover partners are located in different partitions, they won't be able to communicate with each other. Nevertheless, each partner may still be able to serve clients that happen to be part of the same partition.

If this failure mode is taken into consideration, a server can't assume that partner unreachability automatically means that its partner is down. They must consider the fact that the partner may continue operating and interacting with a subset of the clients. The only valid assumption is that the partner also detected the network partition event and follows procedures specified for such a situation.

It should be noted that recovery after a partitioned network is rejoined is significantly more complicated than recovery from a server failure event. As both servers may have kept serving clients, they have two separate lease databases, and they need to agree on the state of each lease (or follow any other algorithm to bring their lease databases into agreement).

This failure mode is more likely (though still rare) in the situation where two servers are in physically distant locations with multiple network elements between them. This is the case in geographically

distributed failover (see Section 4.2).

5.2. Synchronization mechanisms

Partners must exchange information about changes made to the lease database. There are at least two types of synchronization methods that may be used. These concepts are related to distributed databases, so some familiarity with distributed database technology is useful to better understand this topic.

5.2.1. Lockstep

When a server receives a REQUEST message from a client it consults its lease database and assigns requested addresses and/or prefixes. To make sure that its partner maintains a consistent database, it then sends information about a new or just updated lease to the partner and waits for the partner's response. After the response from its partner is received the REPLY message is transmitted to the client.

This approach has the benefit of having a completely consistent lease database between partners at all times. Unfortunately, there is typically a significant performance penalty for this approach as each response sent to a client is delayed by the total sum of the delays caused by two transmissions between partners and the processing by the second partner. The second partner is expected to update its own copy of the lease database in permanent storage, so this delay is not negligible, even in fast networks.

Due to the advent of fast SSD (solid state disk) and battery backed RAM (random access memory) disk technology, this write performance penalty can be limited to some degree.

5.2.2. Lazy updates

Another approach to synchronizing the lease databases is to transmit the REPLY message to the client before completing the update to the partner. The server sends the REPLY to the client immediately after assigning appropriate addresses and/or prefixes and initiates the partner update at a later time, depending on the algorithm chosen. Another variation of this approach is to initiate transmission to the partner, but not wait for its response before sending the REPLY to the client.

This approach has benefit of a minimal impact on server response times, thus it is much better from a performance perspective. However, it makes the lease databases loosely synchronized between partners. This makes the synchronization more complex (and

particularly the re-integration after a network partition event), as there may be cases where one client has been given a lease on an address or prefix of which the partner is not aware (e.g., if the server crashes after sending REPLY to the client, but before sending update information to its partner).

6. DHCPv4 and DHCPv6 Failover Comparison

There are significant similarities between existing DHCPv4 and envisaged DHCPv6 failovers. In particular both serve IP addresses to clients, require maintaining consistent databases among partners, need to perform consistent DNS Updates, must be able take over bindings offered by failed partner, must be able to resume operation after partner is recovered. DNS conflict resolution works on the same principles in both DHCPv4 and DHCPv6.

Nevertheless, there are significant differences. IPv6 introduced prefix delegation [RFC3633] that is a crucial element of the DHCPv6 protocol. IPv6 also introduced the concept of deprecated addresses with separate preferred and valid lifetimes, both being configured via DHCPv6. Negative response (NACK) in DHCPv4 has been replaced with the ability in DHCPv6 to provide corrected response in a REPLY message that differs from a REQUEST.

Also, the typical large address space (close to 2^64 addresses on /64 prefixes expected to be available on most networks) may make managing address assignment significantly different from DHCPv4 failover. In DHCPv4 it was not possible to use a hash or calculated technique to divide the significantly more limited address space and therefore much of the protocol that deals with pool balancing and rebalancing might not be necessary and can be done mathematically. Also, because of the much lower degree of contention for IP addresses, the DHCPv6 failover protocol does not need to be tuned to support rapid reclamation of IPv6 addresses following the loss of a failover peer's database.

However, DHCPv6 Prefix Delegation is similar to IPv4 addressing in terms of the number of available leases and therefore techniques for pool balancing and rebalancing and more rapid reclamation of prefixes allocated by a failed peer will be needed.

7. DHCPv6 Failover Requirements

This section summarizes the requirements for DHCPv6 failover.

Certain capabilities may be useful in some, but not all scenarios.

Such additional features will be considered optional parts of failover, and will be split and defined in separate documents. As such, this document can be considered an attempt to define requirements for the DHCPv6 failover "core" protocol.

The core of the DHCPv6 failover protocol is expected to provide the following properties:

1. The number of supported partners must be exactly two, i.e., there are at most two servers that are aware of a specific lease.

2. For each prefix or address pool, a server must not participate in more than one failover relationship.

3. The defined protocol must support the m-to-1 model (i.e., one server may form more than one relationship), but an implementation may choose to implement only the 1-to-1 model (i.e., everything from one server is backed on another).

4. One partner must be able to continue serving leases offered by the other partner. This property is also sometimes called "lease stability". The failure of either failover partner should have minimal or no impact on client connectivity. In particular, it must not force the client to change addresses and/or change prefixes delegated to it. Lease stability has the aim of avoiding disturbance to long-lived connections.

5. Prefix delegation must be supported.

6. Use of the failover protocol must not introduce significant performance impact on server response times. Therefore synchronization between partners must be done using some form of lazy updates (see Section 5.2.2).

7. The pair of failover servers must be able to recover from a server down failure (see Section 5.1.1).

8. The pair of failover servers must be able to recover from a network partition event (see Section 5.1.2).

9. The design must allow secure communication between the failover partners.

10. The definition of extensions to this core protocol should be allowed, when possible.

Depending on the specific nature of the failure, the recovery

procedures mentioned in points 7 and 8 may require manual intervention.

High Availability is a property of the protocol that allows clients to receive DHCPv6 services despite the failure of individual DHCPv6 servers. In particular, it means the server that takes over providing service to clients from its failed partner, will continue serving the same addresses and/or prefixes. This property is also called "lease stability".

Although progress on a standardized inter-operable DHCPv4 failover protocol has stalled, vendor-specific DHCPv4 failover protocols have been deployed that meet these requirements to a large extent. Accordingly it would be appropriate to take into account the likely coexistence of DHCPv4 and DHCPv6 failover solutions. In particular, certain features that are common to both IPv4 and IPv6 implementations, such as DNS Update mechanism, should be taken into consideration to ensure compatible operation.

7.1. Features out of scope

The following features are explicitly out of scope.

1. Load Balancing - a capability is considered an extension and may be defined in a separate document. It must not be part of the core protocol, but rather defined as an extension. The primary reason for this the desire to limit core protocol complexity. Load Balancing is likely to be defined as an extension. See [I-D.ietf-dhc-dhcpv6-load-balancing].

2. Configuration synchronization - two failover partners are expected to maintain the same configuration. Mismatched configuration between partners is a frequent problem in failover solutions. Unfortunately, that is an open-ended problem, since different servers have very different configuration data models.

3. m-to-n model (see Section 4.4)

4. Servers participating in multiple failover relationships for any given prefix or address pool.

8. Security Considerations

The design must provide a mechanism whereby each peer in a failover relationship can identify the other peer, authenticate that identification, and validate that the identified peer is the one with which communication is intended. This mechanism should also

optionally provide support for confidentiality.

The protocol specification, when it is written, should provide operational guidelines in the case of authentication mechanisms that require access to network servers that have the potential to be unreachable (e.g. what to do if a partner is reachable, but remote Certificate Authority is unreachable due to network partition event).

The security considerations for the design itself will be discussed in the design document.

9. IANA Considerations

IANA is not requested to perform any actions at this time.

10. Acknowledgements

This document extensively uses concepts, definitions and other parts of [dhcpv4-failover] document. Thanks to Bernie Volz and Shawn Routhier for their frequent reviews and substantial contributions. Authors would also like to thank Qin Wu, Jean-Francois Tremblay, Frank Sweetser, Jiang Sheng, Yu Fu, Greg Rabil, Vithalprasad Gaitonde, Krzysztof Nowicki, Steinar Haug, Elwyn Davies, Ted Lemon, Benoit Claise and Stephen Farrell for their comments and feedback.

This work has been partially supported by Department of Computer Communications (a division of Gdansk University of Technology) and the National Centre for Research and Development (Poland) under the European Regional Development Fund, Grant No. POIG.01.01.02-00-045 / 09-00 (Future Internet Engineering Project).

11. References

11.1. Normative References

[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.

[RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.

[RFC4704] Volz, B., "The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Client Fully Qualified Domain Name (FQDN)

Option", RFC 4704, October 2006.

11.2. Informative References

[I-D.ietf-dhc-dhcpv6-load-balancing]
Kostur, A., "DHC Load Balancing Algorithm for DHCPv6", draft-ietf-dhc-dhcpv6-load-balancing-00 (work in progress), December 2012.

[RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.

[RFC5970] Huth, T., Freimann, J., Zimmer, V., and D. Thaler, "DHCPv6 Options for Network Boot", RFC 5970, September 2010.

[RFC6853] Brzozowski, J., Tremblay, J., Chen, J., and T. Mrugalski, "DHCPv6 Redundancy Deployment Considerations", BCP 180, RFC 6853, February 2013.

[dhcpv4-failover]
Droms, R., Kinnear, K., Stapp, M., Volz, B., Gonczi, S., Rabil, G., Dooley, M., and A. Kapur, "DHCP Failover Protocol", draft-ietf-dhc-failover-12 (work in progress), March 2003.

Authors' Addresses

Tomek Mrugalski
Internet Systems Consortium, Inc.
950 Charter Street
Redwood City, CA 94063
USA

Phone: +1 650 423 1345
Email: tomasz.mrugalski@gmail.com

Kim Kinnear
Cisco Systems, Inc.
1414 Massachusetts Ave.
Boxborough, Massachusetts 01719
USA

Phone: +1 (978) 936-0000
Email: kkinnear@cisco.com