

VitalAAA IPAMv2 Quick start guide

Overview

The IPAMv2 subsystem is an IP Address Pool and address allocation manager tightly coupled to the session authorization mechanism defined by the AAA protocols supported by VitalAAA, i.e. RADIUS, Diameter and TACACS+. IPAMv2 provides allocation of both IPv4 addresses as well as arbitrarily wide IPv6 prefixes.

The RADIUS, Diameter and TACACS+ protocols all define an abstraction of client state tracking on the server side through a mechanism known as sessions. In RADIUS for example, the server creates a session object whose state is set to "waiting for start" for a client when an authentication request is received. The session is later brought into "active" state when the client sends an accounting-*Start* request. When the client decides to relinquish resources and an accounting-*Stop* request is sent, the session object's state is set to "inactive".

The lifetime semantics of IP address allocations are identical to those of AAA sessions and thus fits in perfectly with the AAA session model. By integrating IP address management with other AAA resource allocation several benefits are gained:

- IP allocation state is tied in with the AAA session state without the time-lag associated with separate allocation servers.
- All session parameters are managed from by single point of configuration and provide a single source of status.
- Overcomes many of the weaknesses of DHCP, such as over-allocation of addresses from lack of client-releases, poor support of IPv6 prefixes and the typical limitation of client identification (usually a 48-bit MAC address).
- High availability support through existing AAA server technology (HA-USS – the Universal State Server, or “session database”).

IPAMv2 is technically a separate subsystem in VitalAAA with its own allocation scheme, but is tightly integrated with the USS and is available in PolicyFlow through the StateServer Plugin.

IP Address allocation and de-allocation requests are fed to IPAMv2 as a result of session state changes observed within the USS. Several fields of data necessary for IPAMv2 operations are maintained in the USS session record. These fields are visible as part of the entry from the Administrative Interface and are replicated between the servers in HA-USS setups.

In addition to the normal replication and persistence scheme provided by the USS, upon every state change IP allocations are maintained in a dedicated

lease-database on hard-disk. The use of persistent storage is designed to minimize any chance of loss of the allocation state, and provides good integrity even for a standalone server.

Provisioning

There are two mechanisms available to provision IPAMv2; The SMT (Server Management Tool – The VitalAAA graphic management interface) and LDAP. The LDAP interface is described in a separate document. This document describes the basic provisioning which, coupled with the SMT GUI, should make the use of the SMT intuitive.

As previously noted, the USS initiates allocation request through use of the StateServer Plugin. For the purpose of the discussion in this document, the following excerpt from a PolicyFlow is used as a reference:

```
checkLocalLimits Method-Type = StateServer
  StateServer-RequestMap = "${uss.*} := ${request.*};"
  StateServer-RequestMap = "${uss.User-Name} := ${packet.Base-User-Name};"
  StateServer-RequestMap = "${limit.User-Name} := ${packet.Limit-User-Name:1};"
  StateServer-RequestMap = "${uss.Full-User-Name} := ${request.User-Name};"
  StateServer-ReplyMap = "${reply.framed-ipv6-prefix} := ${ipam.prefix};"
  StateServer-ReplyMap = "${reply.framed-ip-address} := ${ipam.address};"
  StateServer-PoolSelector = "${request.va-string-0}"
```

Pool selectors

IPAMv2 allocation is enabled by populating the *StateServer-PoolSelector* property with a name of a *pool selector*, in the example this is stored in *request.va-string0*. A pool selector refers to an ip-pool or set of *ip pools*, and can be thought of as an intermediary in the allocation process. Each pool selector can refer to any number of ip pools of the same type of allocation unit, i.e. all must be of type IPv4 or of type IPv6 with the same prefix width.

There must be no overlapping addresses among the pools in a pool-selector, although overlap between separate pool selectors and ip pools is perfectly valid.

The relation between ip pools and pool selectors is N-N, i.e. not only can a pool selector contain any number of pools, but a pool can appear in any number of pool selectors. It is strongly recommended that caution be used when setting up the pool selectors to avoid unexpected behavior from overly complex relations.

IP pools in a selector are currently utilized exhaustively, all addresses are used up in the first pool before allocation from the next is started. The ordering in which pools within a selector are selected is currently random, but user-controls to change the scheme may be exposed in the future.

IP pools

Each IP pool has a number of attributes related to the pool including an arbitrarily large set of inclusion and exclusion ranges. No range is allowed to exceed 16M prefixes/addresses, but any number of ranges can be defined in a pool. Each prefix/address in an inclusion range that is not covered by an exclusion range will

consume server resources, mainly memory foot-print, so care must be taken to not define outrageously large pools.

As an example, the following include range definitions would each consume 16M IP Addresses/prefixes:

```
6278:6ae3::/64-6278:6ae3:ff:ffff::/64
```

```
135.0.0.0-135.255.255.255
```

The following IP Pool attributes may be defined for an IP Pool through the SMT GUI:

GUI Label	Variable Name	Type	Explanation
Include	inc-range	String	High and low address or prefix value separated by a dash. Multiple Include Ranges may be defined for a pool.
Exclude	exc-range	String	High and low address or prefix value separated by a dash. Multiple Exclude Ranges may be defined for a pool.
Active	active	boolean	Set to false deactivates the pool, i.e. no allocation is made from the pool. Deactivating an active pool will not affect outstanding allocations.
Low Water	low-water	integer %	See below.
High Water	high-water	integer %	See below.
Assign Method	assign-method	"mru" or "lru"	Setting assign-method to "mru" causes the last released allocation to be reallocated first and setting it to "lru" caused the last released allocation to be reallocated last.
Lease-time	lease-time	integer	Attribute whose value is available in the StateServer Plugin as ipam.lease-time for use by the PolicyFlow writer, possibly to be used to control session life-time.
Reuse Time	reuse-time	integer seconds	Guard time during which an allocation is kept in an intermediate "reserved" state before becoming available after getting de-allocated.
Orphan Holdoff	orphan-holdoff	integer seconds	See below.

Identity	identity	integer, read-only	Unique identity assigned every time a new pool is created. The id is the search key used internally by the IPAM to locate a pool in a selector. Hence, deleting and recreating a pool with the same name does <i>not</i> make it available in the selector it was previously referenced.
Debug Level	debug-level	integer	Not currently used.

High and Low Water marks

The high and low water marks are used by the IPAM to control log-events. Both values are defined as integer percentage values between 0% and 100% of the utilization of a given ip pool. The IPAM translates the two water-marks to absolute numbers of addresses as

$$(\text{water mark}) * (\text{total addresses}) / 100$$

The first time the number of allocated addresses exceeds the high water value a high-water log-message is issued.

The low water mark functions as the reset point after a high-water mark has been exceeded. After a high water-mark has been exceeded, no further high-water mark messages will be logged until the ip pool usage level has dropped below the low-water mark. A low-water log-message is issued the first time pool utilization descends below the low-water level after a high-water log message, and is not reissued until after a new high-water log message has been logged.

The log-messages are issued under the log-area *ipam2.poolutilization* as

```
Pool <name> below low water mark
Pool <name> above high water mark
```

Where <name> is the name of the pool, and can be redirected through the log-director to produce an SNMP trap or any other action available through the VitalAAA logging system.

reuse-time

Specifies the time delay between when an address is de-allocated and when it is marked as "free" for re-assignment. By specifying a positive value for the reuse-time attribute, allocations are put into an intermediate "reserved" state for the *reuse-time* after de-allocation instead of immediately becoming available for allocation.

orphan-holdoff attribute

Session address allocation information is saved to an on-disk database as its state changes. If the VitalAAA server is brought down abruptly without letting it save its USS session data, for example, because of a power failure, and is

subsequently brought up, the lease information will be recovered from the on-disk database.

Any sessions that are determined to be in an allocated state will then be installed as "orphaned" USS session entries. The USS entryfields will be populated from the information available in the lease-file. The orphan-holdoff attribute specifies the timeout such entries are given before they are automatically placed into an inactive state. Two special values exist for orphan-holdoff: -1 means infinity, i.e. never time out, and 0 means do not install "orphaned" entries at all. The Ip Address allocation information kept in the lease-database and its use is as follows:

Field Name	Use
creation time	The time the lease-entry was created. This value is only available through IPAM commands, see below.
modify time	The last time the entry was persisted. This value is only available through IPAM commands, see below.
Prefix	The IP address or prefix, installed in the USS entry.
State	The state of the lease, "allocated" if the lease was in use when it was persisted, otherwise "reserved" or possibly "available". Allocated leases are subject to be installed as USS entries based on the value of orphan-holdoff.
Owner	The USS owner, e.g. NAS+Port.
NAS	The NAS from the USS entry.
Session	The session-id used to allocate the entry.

Note: The sum of the lengths of Owner, NAS and Session, encoded in UTF-8, may not exceed 95 bytes.

If a new client session information is received that matches the USS key of an orphaned entry, the USS will process the record through the normal state transitions, i.e. be reallocated if the request is a start or de-allocated if the request is a stop and so on.

PolicyFlow and USS specifics

Server properties

No server properties are defined for the IPAMv2 when it is running in standalone mode.

When the USS is running in fault-tolerant mode (using the HA-USS), the *StateServer-SecondaryAddress* server property is required in addition to the usual *StateServer-ReplicationRole*. This property must be set to the address and port of the secondary server to enable replication of provisioning information, i.e. ip pool and pool selector provisioning.

Provisioning an HA-USS setup can only be done on the primary server when the secondary server is up and reachable. It is *not* possible to provision an active secondary during a fail-over condition.

PolicyFlow

After an IP Address allocation has been made, the prefix and/or address is available in an IPAMv2 variable to be used as a property of the StateServer Plugin (as *ipam.prefix* and *ipam.address* - see PolicyFlow example above). Depending on whether the value is to be used as a PolicyFlow prefix variable or an address variable, the writer will choose one or the other.

More specifically, the ipam-prefix to PolicyFlow value conversion matrix is as follows:

Type	Width	IPv4AddressValue	IPv6AddressValue	IPv6PrefixValue
v4	32	yes	no	no
v6	128	no	yes	yes
v6	<128	no	no	yes

There are three other IPAMv2 internal variables of interest:

ipam.leasetime which contains the value given to the ip pool currently selected,

ipam.pool-selector which contains the pool-selector value

ipam.pool-name which contains the selected pool's name.

These are read-only variables used internally in the IPAMv2 and may be read via the StateServer plug-in *StateServer-ReplyMap* property. For example:

```
StateServer-ReplyMap = "${reply.framed-ip-address} := ${ipam.address};"
```

USS Session Entries

Besides their usual content, USS session entries are populated with the current IP address/prefix, the numeric pool-id, the IP pool's name and the name of the selector used when the allocation was made.

All this information as well as the additional information from the lease-entry listed above is saved to the persistent lease database during clean server shutdowns and is replicated between the servers in HA-USS configurations.

Admin Interface (AI) commands

Two AI commands are available to query for IPAMv2 information;

ipam lease and *ipam pool*.

The first command, *ipam lease*, takes a single IP address and returns the IPAMv2 entry for that address.

```
ipam lease [pool-selector] <IP Address>
```

The IP Address is required. The optional pool-selector name limits the search for the IP Address to the specified pool-selector. For example, the command

```
ipam lease 192.168.4.25
```

returns lease information for all occurrences of the IP address 192.168.4.25 in the IPAMv2. The command

```
ipam lease selector-1 192.168.4.25
```

returns lease information for an occurrences of the IP address 192.168.4.25 in all pools contained in pool selector "selector-1."

The command *ipam pool* lets the user view the entire contents of a given pool. Options allow listing of only allocated addresses/prefixes, only free addresses/prefixes, or all addresses/prefixes.

```
ipam pool <pool-name> <all|used|free> [filename]
```

Due to the potentially very large output of the command, the command takes an optional file-name argument to specify a file to which the command output is saved instead of printing to the AI terminal. Files are written to the server's run-directory. As a safety precaution, the command will not overwrite existing files.