NETWORK SYSTEMS AUTOMATION ON A SHOESTRING BUDGET

OR ALTERNATIVELY,

HOW I TURNED LOTS OF LITTLE SCRIPTS INTO A COUPLE OF BIG ONES.

AYDEN BEESON, CHARLES STURT UNIVERSITY

A LITTLE BIT ABOUT ME

- Started at CSU as a student in 2005, studying my B. InfoTech.
- Started working at CSU in 2007, as a local campus network and PABX tech working with our old Ericsson MD110 PABXs
- Removed the old Modem pools from the campus, as well as decommissioned our Microwave WAN links in favour of AARNET fibre links.
- Oversaw the replacement and decommissioning of the PABX's with our current IP based telephony system
- Over the years, have become one of the senior network staff
- I have 3 kids, all currently at primary school, 2 cats and 1 dog.

SOME QUICK HISTORY

THE PREVIOUS SYSTEM - DNS

- DNS provided by BIND, with static configuration files
- Zones created and deleted by hand
- Updates to TLD csu.edu.au done by hand
- Updates to reverse zones and forward AD domain done by “chase” script via DHCP

THE PREVIOUS SYSTEM – DHCP

- DHCP provided by ISC DHCPD
- Configuration pushed via netreg
- Hosts file exported and uploaded via Netreg
- Chase script watching DHCP logging, and inserting DNS entries as machines are detected
- No removal mechanism for entries
  - forward stays until overwritten by the device getting a new IP
  - reverse stays until another device takes that specific IP

THE PREVIOUS SYSTEM - NETREG

- Netreg generating DHCP configuration based on a few different template files
- Some limited DNS capabilities
- IP addresses tracked and duplicates prevented, but only used for DHCP fixed addressing
- No subnet tracking, grouping, hierarchy etc.
- IPAM done via a spreadsheet, which by this point had about 10 tabs with the different private spaces, public spaces, planned vs implemented, VLANs etc etc.

NETREG – DHCP CONFIGURATION

- Written only to handle our public IP space, with fixed /24 subnets
- Hacked later on to handle private IP ranges
- Hacked again to handle phone boot strings and options
- Hacked again to handle wireless boot strings and options, including devel settings
- Hacked again to handle non classful subnets (/22s)

NETREG – DHCP UPDATES

- Registered devices exported into hosts file for DHCP to use
- Fixed addresses exported for use by DHCP
- Secondary mac addresses (Wi-Fi cards, Ethernet Dongles etc) exported for use as well

NETREG – DNS UPDATES

- Some triggered DNS update mechanisms for certain devices, generally consisting of nsupdate entries via
  - chase script
  - DB flag and regular script push
- No DNS zone or configuration management

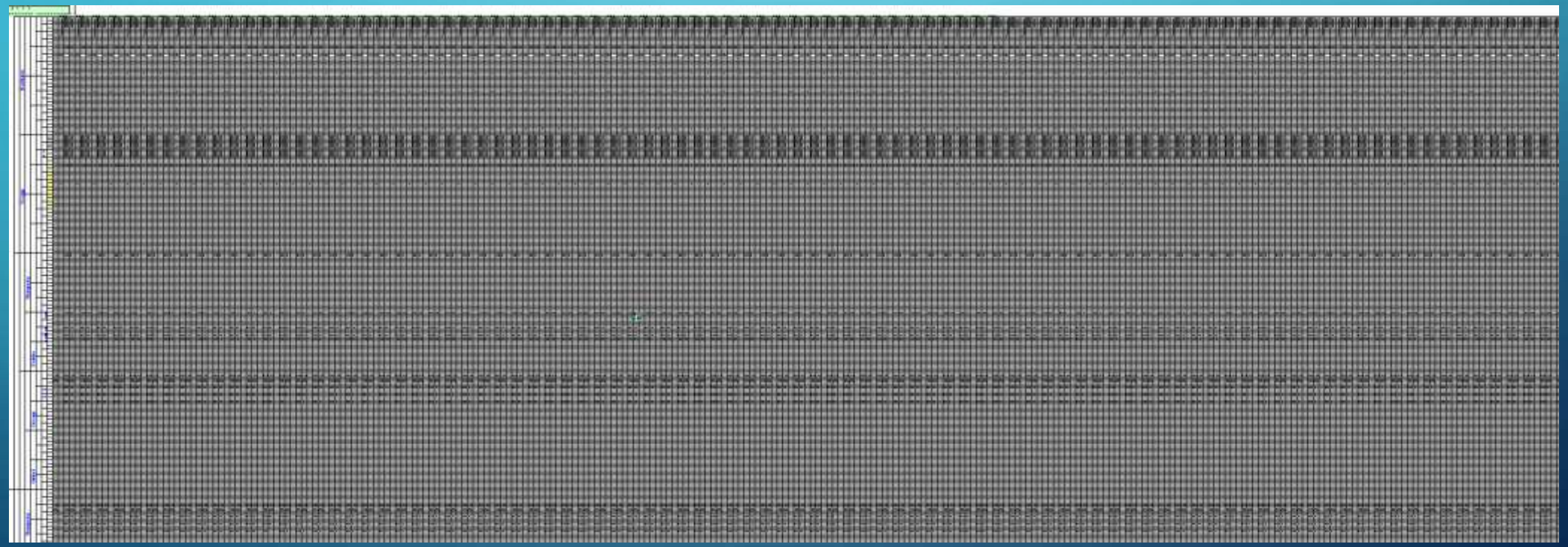
THE STRAW THAT BROKE THE CAMELS BACK

IPv6

WHILE IPV4 WORKS OK IN A SPREADSHEET

137.166.x.x		Charles Sturt University (Planned) (3rd Octet)							
65536	32768	16384	8192	4096	2048	1024	256		
1	2	4	8	16	32	64	256		
65534	32766	16382	8190	4094	2046	1022	254		
65534	65532	65528	65520	65504	65472	65408	65024		
/16	/17	/18	/19	/20	/21	/22	/24		VLAN name
0	128	192	224	240	248	252	255		(class C size)
						Mgmt & Server	Management	0	
							Mgmt & Loopback	1	
							Point-to-Point	2	
							BSEC3	3	bsec3
						Server	BPUB4	4	bpub4
							BCAMP5	5	bcamp5
							Server (Future)	6	
							Server (Future)	7	
					Radio		BRAD8	8	brad8
							BRAD9	9	brad9
							BRAD10	10	brad10
							BRAD11	11	brad11
							BRAD12	12	brad12
							BRAD13	13	brad13
							BRAD14	14	brad14
							BRAD15	15	brad15
					Staff		BSTF16	16	bstf16
							BSTF17	17	bstf17
							BSTF18	18	bstf18
							BSTF19	19	bstf19
							BSTF20	20	bstf20
							BSTF21	21	bstf21
							BSTF22	22	bstf22
							BSTF23	23	bstf23

TURNS OUT IPV6 DOES NOT….



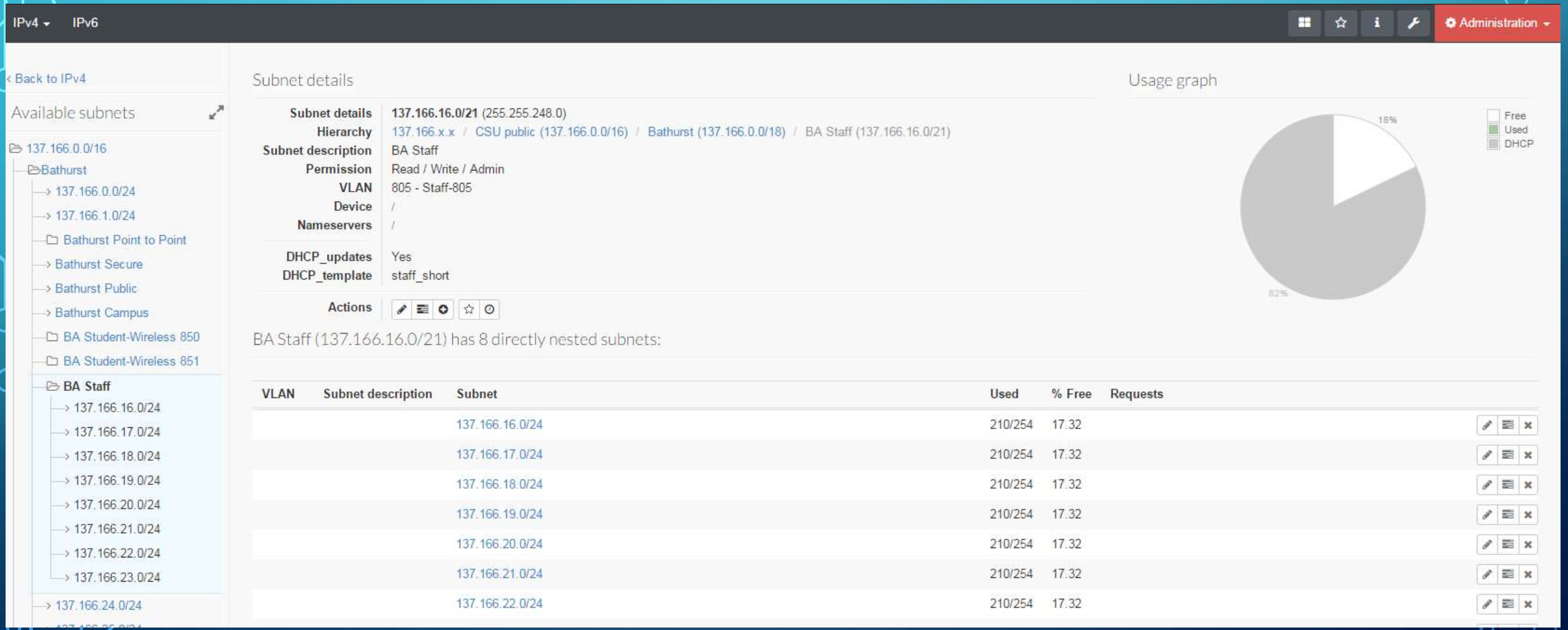
WE STARTED LOOKING AT OPTIONS

- Lots of commercial IPAM solutions out there, with varying degrees of DNS / DHCP integration
- Most rely on either hardened Linux appliances, or windows servers to do the work
- Costly and/or very much a closed ecosystem

IPAM “BUDGET”

- We had flagged a potential spend on an IPAM solution, but given IPv6 was essentially an experiment I was pushing, it was not a priority item
- When I said shoestring, I really mean “zero”

ENTER PHPIPAM



PHPIPAM HAS GIVEN US:

- Clear hierarchical layout of our IPv4 and IPv6 space
- The ability to easily see what subnets were allocated where
- VLAN tracking linked to subnets
- Device ownership for both subnets and individual IPs
- Percentage counts on IP utilisation
- An **active** open source product, github listed and regularly maintained
- And most importantly, it was free!

SO WHAT NOW?

- We have an IPAM solution, but its isolated.
- Just a records tool, really no better than the spreadsheet
- Static data stores have a tendency to get out of date / neglected
- We still haven't fixed our DHCP or DNS problems

TIME TO INTEGRATE!

Solution needed to:

- Control DHCP configuration
- Control DNS configuration
- Allow for static hosts, both in DNS and as fixed addresses in DHCP
- Allow for dynamic DNS from DHCP clients
- Be easy to use
- As automatic as possible
- As fast as possible

DHCP AND DNS

- We were happy with ISC DHCPD and ISC BIND
- They had been running for years and were well understood
- DHCPD had a well developed failover mechanism
- Dynamic DNS updates supported natively from DHCPD using BIND
- Allowed us to use standard RHEL VMs to host services
- Site license for RHEL + VM's makes it very easy

DNS ZONE MANAGEMENT

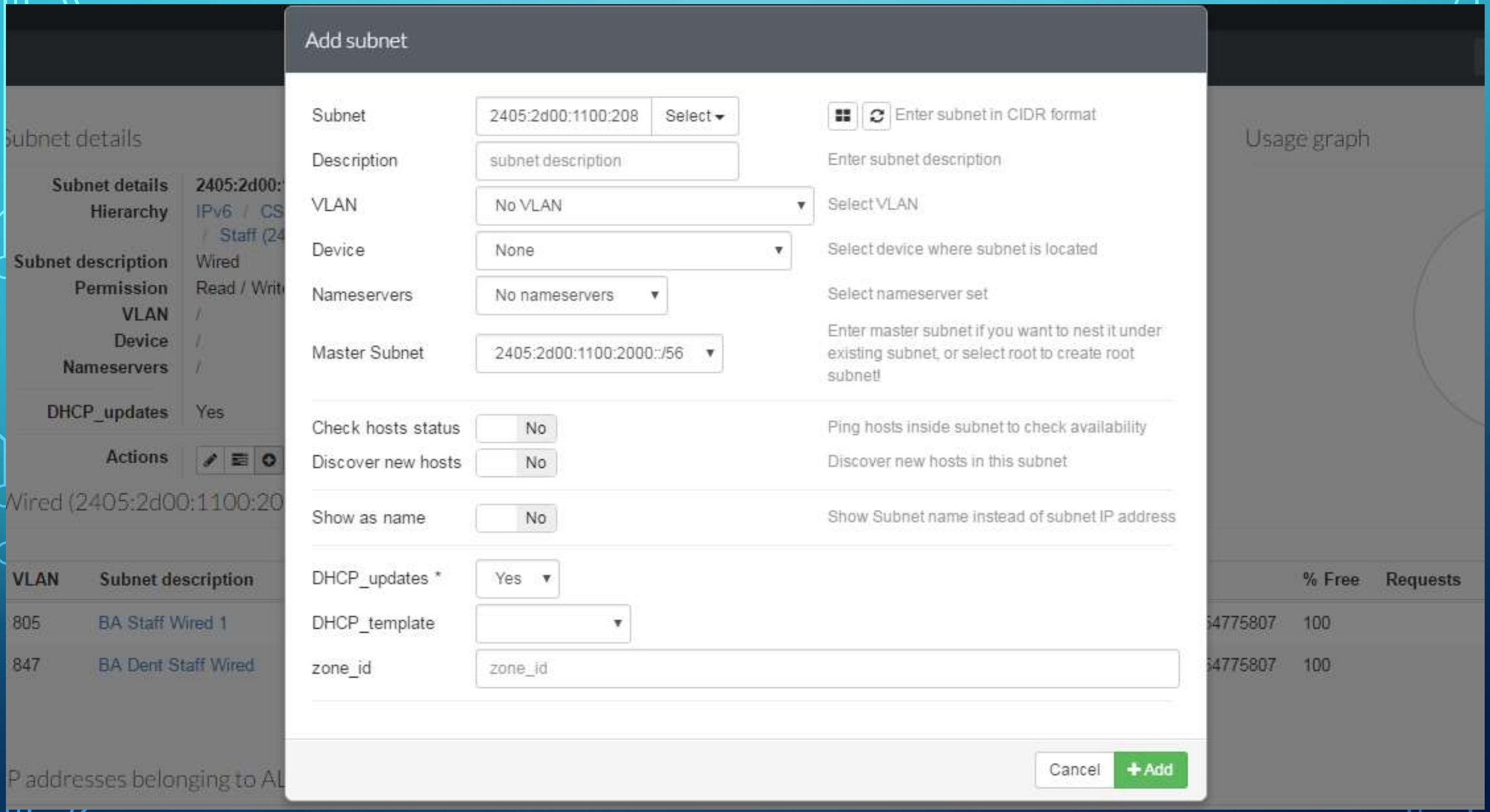
After deciding not to start from scratch for BIND, we found NicTool

- Open source, GitHub listed and actively maintained
- Written in Perl
- MySQL DB backend, same as PHPIPAM
- Can configure and control BIND (only partially useful to us)
- Free too!
- Interface is a little old, but functional

SO WHAT DOES IT DO?

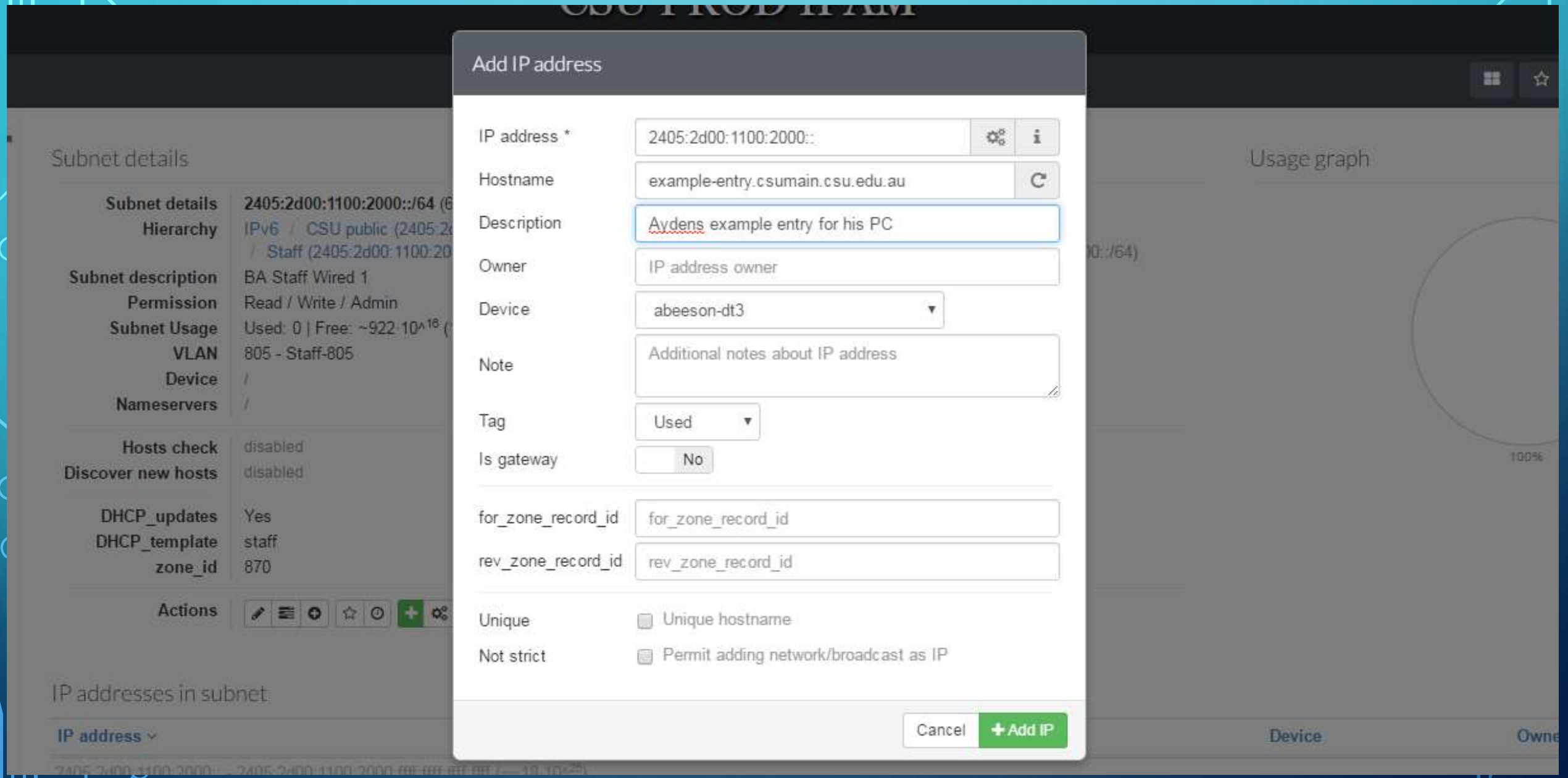
ADDING A SUBNET

- Using IPAM, you add the subnet into the hierarchy where it is needed
- If the subnet is a /24 (or a /64 for IPv6), a matching zone is created within NicTool to match the subnet
- If DHCP is required, a template is added to the subnet
- The subnet is then included in the next DHCPD export to be created when puppet is run on the DHCPD servers
- The template determines how big the pools are (if any)

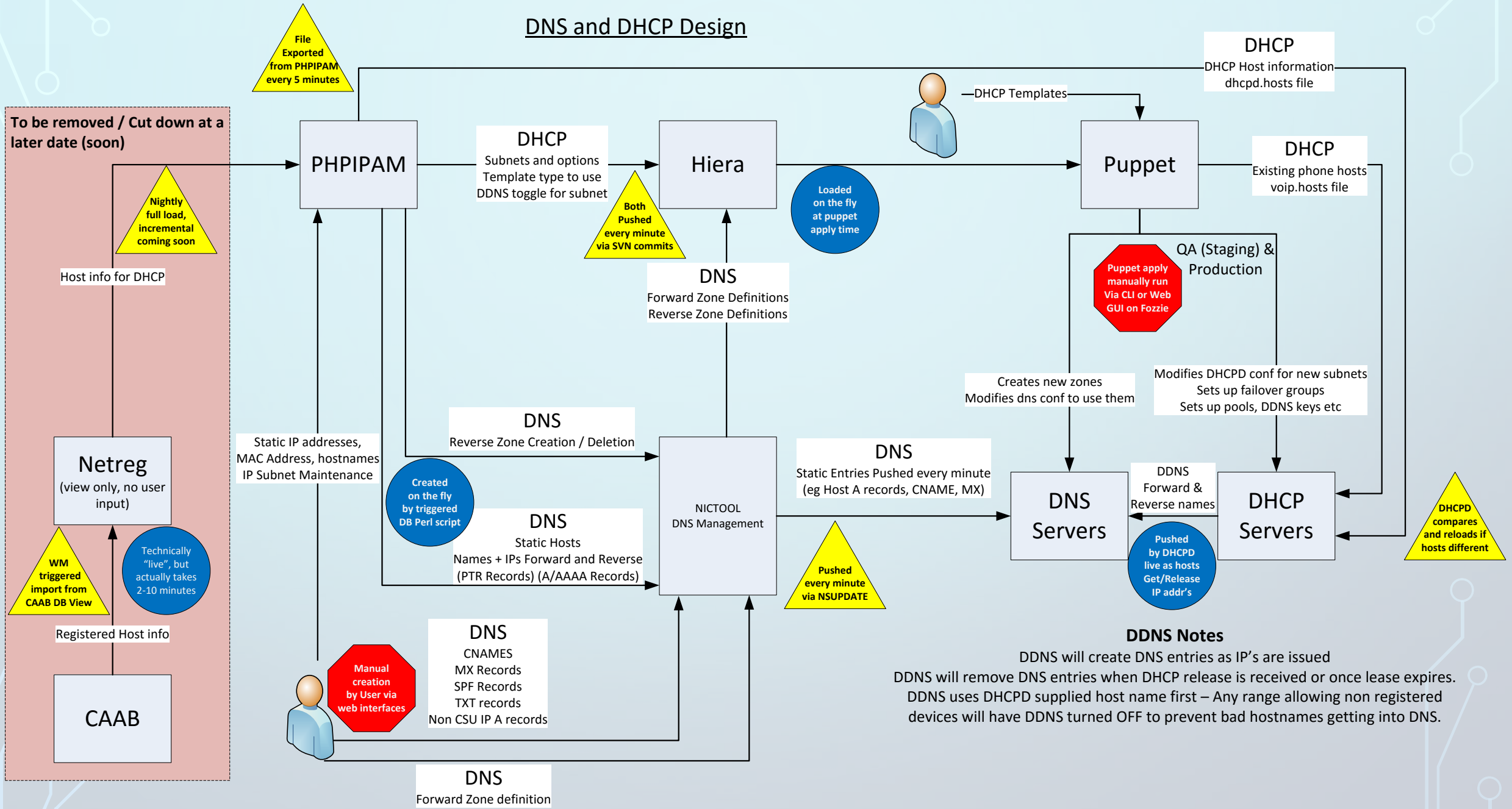

Add subnet
Subnet
2405:2d00:1100:208
Select
Enter subnet in CIDR format
Description
subnet description
Enter subnet description
VLAN
No VLAN
Select VLAN
Device
None
Select device where subnet is located
Nameservers
No nameservers
Select nameserver set
Master Subnet
2405:2d00:1100:2000::/56
Enter master subnet if you want to nest it under existing subnet, or select root to create root subnet!
Check hosts status
No
Ping hosts inside subnet to check availability
Discover new hosts
No
Discover new hosts in this subnet
Show as name
No
Show Subnet name instead of subnet IP address
DHCP_updates *
Yes
DHCP_template
zone_id
zone_id
Cancel
+ Add

ADDING A HOST

- Using PHPIPAM, you create a device using a name and mac address
- If the host needs static DNS or a fixed address, you assign it against an IP within the required subnet, providing a fully qualified host name
- The hostname is pushed into NicTool and added to the required reverse and forward zones. This entry is then pushed into DNS automatically on the next minute.
- The host details are pushed by PHPIPAM into the DHCPD hosts file, which is sent to the DHCPD servers within the next 5 minutes and loaded automatically


Add IP address
IP address *
2405:2d00:1100:2000::
Hostname
example-entry.csumain.csu.edu.au
Description
Aydens example entry for his PC
Owner
IP address owner
Device
abeeson-dt3
Note
Additional notes about IP address
Tag
Used
Is gateway
No
for_zone_record_id
for_zone_record_id
rev_zone_record_id
rev_zone_record_id
Unique
Unique hostname
Not strict
Permit adding network/broadcast as IP
Cancel
Add IP
Subnet details
Usage graph
IP addresses in subnet
IP address
Device

DNS and DHCP Design



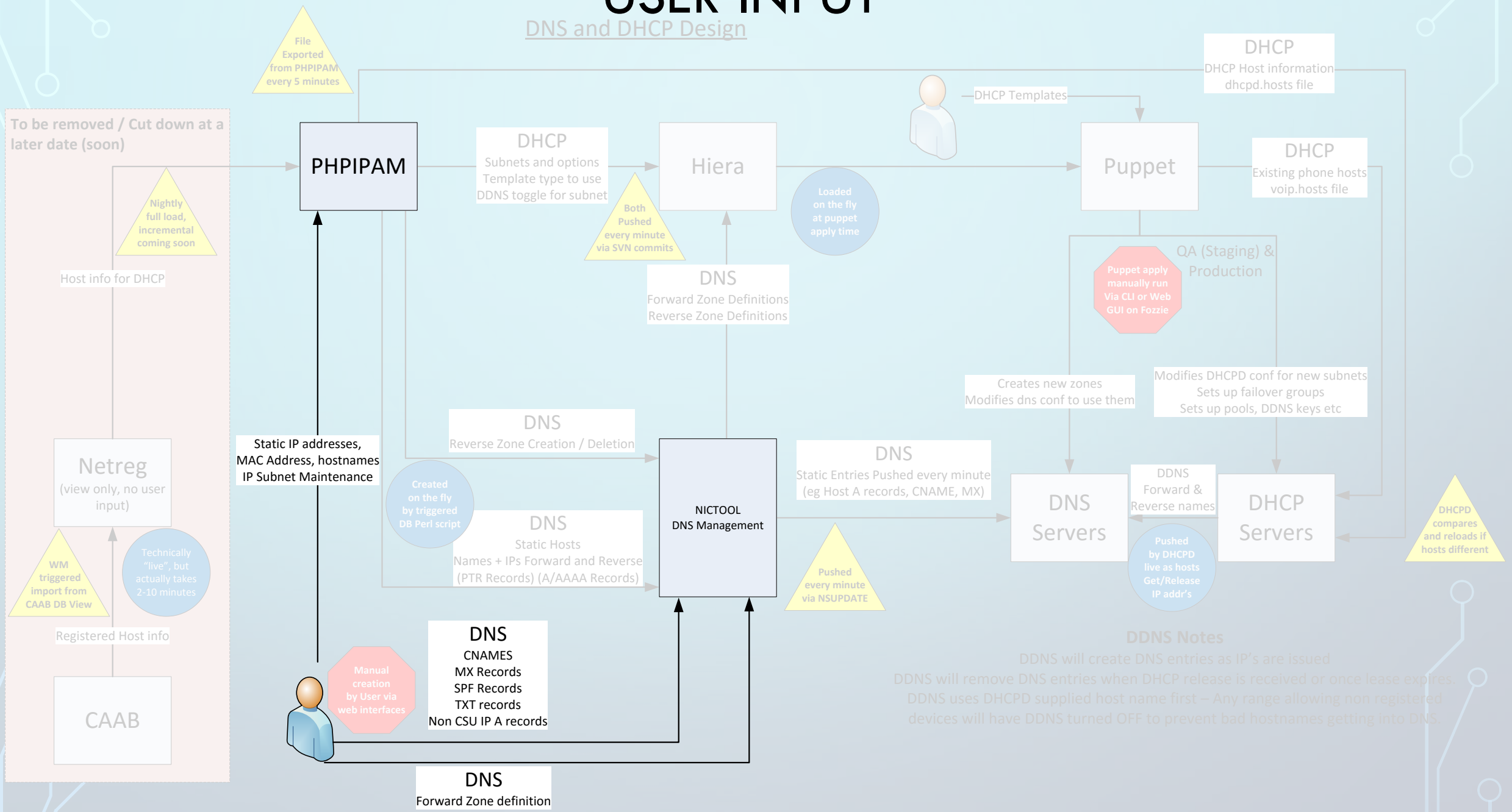
DDNS Notes

DDNS will create DNS entries as IP's are issued

DDNS will remove DNS entries when DHCP release is received or once lease expires.

DDNS uses DHCPD supplied host name first – Any range allowing non registered devices will have DDNS turned OFF to prevent bad hostnames getting into DNS.

USER INPUT

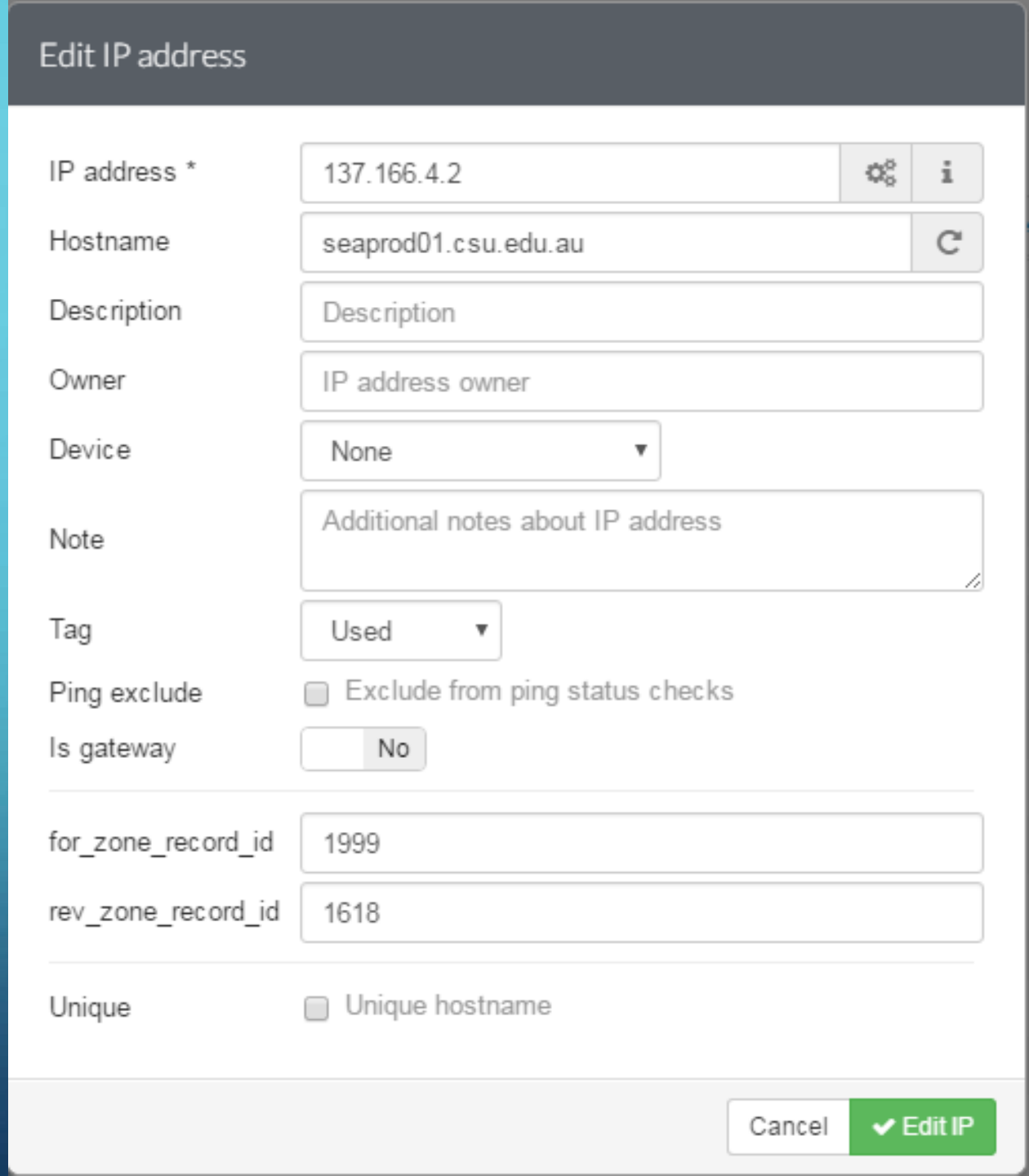

DNS and DHCP Design
File Exported from PHPIPAM every 5 minutes
DHCP
DHCP Host information
dhcpd.hosts file
DHCP Templates
To be removed / Cut down at a later date (soon)
PHPIPAM
DHCP
Subnets and options
Template type to use
DDNS toggle for subnet
Hiera
Puppet
DHCP
Existing phone hosts
voip.hosts file
Nightly full load, incremental coming soon
Both Pushed every minute via SVN commits
Loaded on the fly at puppet apply time
Host info for DHCP
DNS
Forward Zone Definitions
Reverse Zone Definitions
Puppet apply manually run Via CLI or Web GUI on Fozzie
QA (Staging) & Production
Creates new zones
Modifies dns conf to use them
Modifies DHCPD conf for new subnets
Sets up failover groups
Sets up pools, DDNS keys etc
Static IP addresses, MAC Address, hostnames IP Subnet Maintenance
DNS
Reverse Zone Creation / Deletion
Created on the fly by triggered DB Perl script
DNS
Static Hosts
Names + IPs Forward and Reverse
(PTR Records) (A/AAAA Records)
NICTOOL
DNS Management
DNS
Static Entries Pushed every minute
(eg Host A records, CNAME, MX)
DNS Servers
DDNS
Forward & Reverse names
DHCP Servers
Pushed by DHCPD live as hosts Get/Release IP addr's
DHCPD compares and reloads if hosts different
Netreg
(view only, no user input)
WM triggered import from CAAB DB View
Technically "live", but actually takes 2-10 minutes
Registered Host info
CAAB
Pushed every minute via NSUPDATE
Manual creation by User via web interfaces
DNS
CNAMES
MX Records
SPF Records
TXT records
Non CSU IP A records
DNS
Forward Zone definition
DDNS Notes
DDNS will create DNS entries as IP's are issued
DDNS will remove DNS entries when DHCP release is received or once lease expires
DDNS uses DHCPD supplied host name first – Any range allowing non registered devices will have DDNS turned OFF to prevent bad hostnames getting into DNS.

USER INPUT

- User input enters the system into both PHPIPAM and NicTool
- Both systems are web accessible and run on the same host
- Apache runs to serve the web content for both systems
- PHPIPAM supports AD Auth and has been integrated into our standard authentication systems. Users are directly added to authorise them and set their groups as required
- NicTool has the capability to have LDAP logins, but it's a fairly new feature and we haven't had a chance to get it going yet.

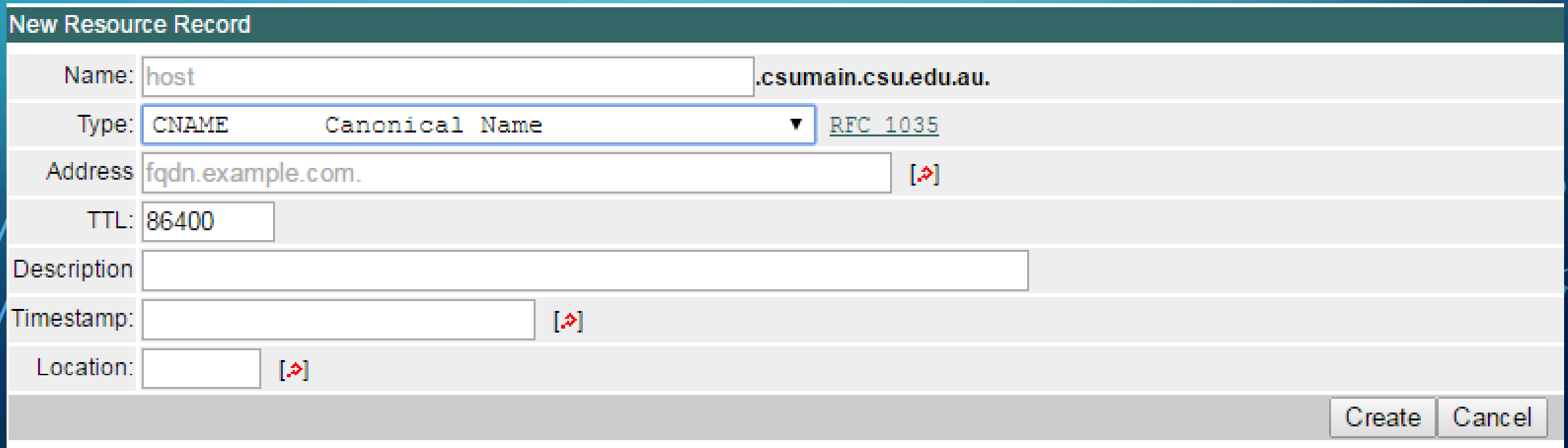
USER INPUT - PHPIPAM

- PHPIPAM is used to create subnets and assign DHCP templates
- PHPIPAM is also used to assign static IP's to devices, and to set fixed DNS names for anything that doesn't DHCP, or needs static DNS



USER INPUT - NICTOOL

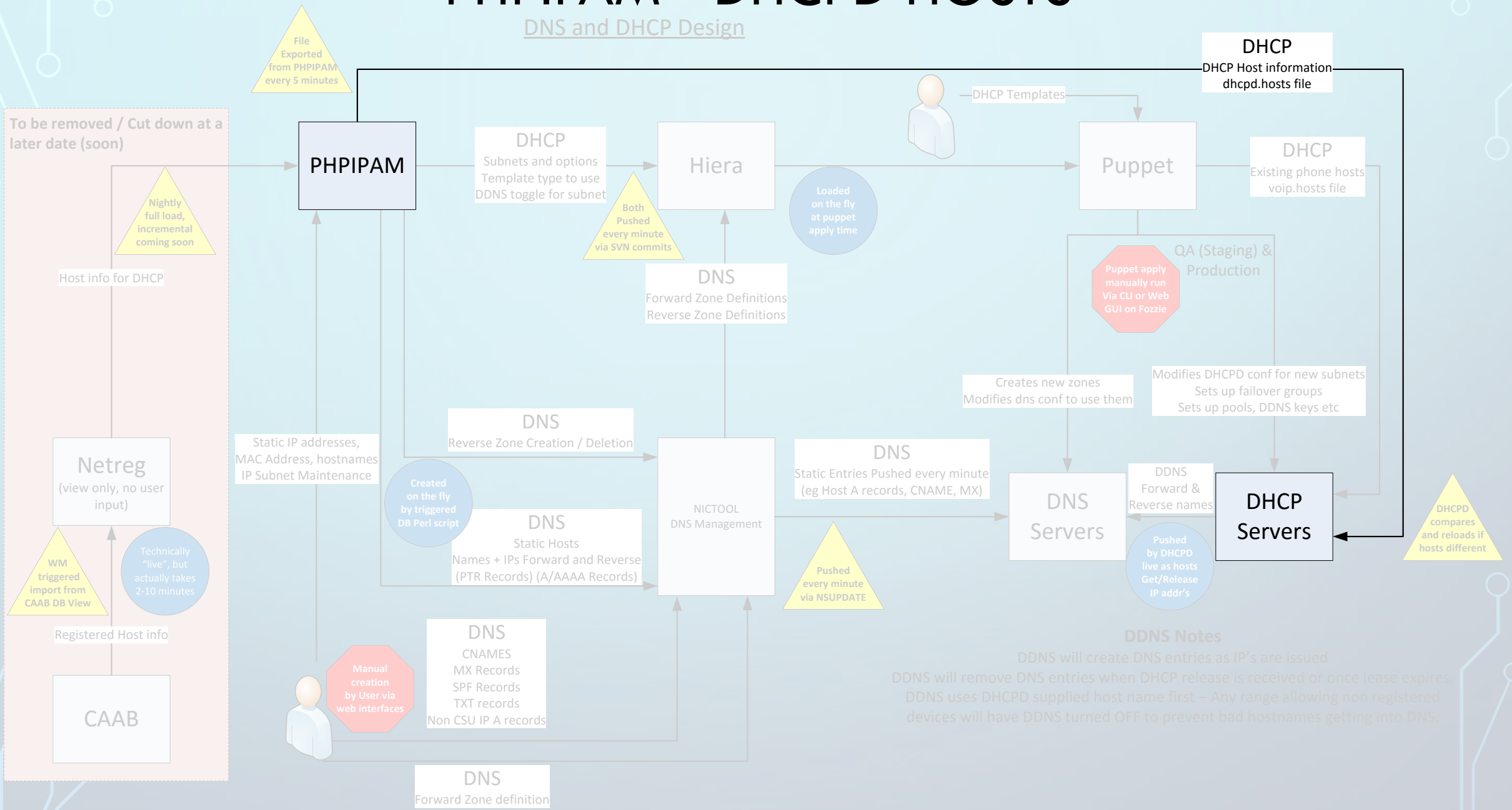
- NicTool gets most of the data from PHIPAM, however some records cannot be tied to an IP directly
- CNAMES, MX records, NS records and SPF records are all added into NicTool directly



USER INPUT – NICTOOL

- We have a blanket ban on A records being assigned manually, ensuring IPAM stays accurate and IP's are "owned" directly
- One unfortunate exception for this was our SMTP mail host entry, which didn't like CNAME's and had to have a round robin A record of its own

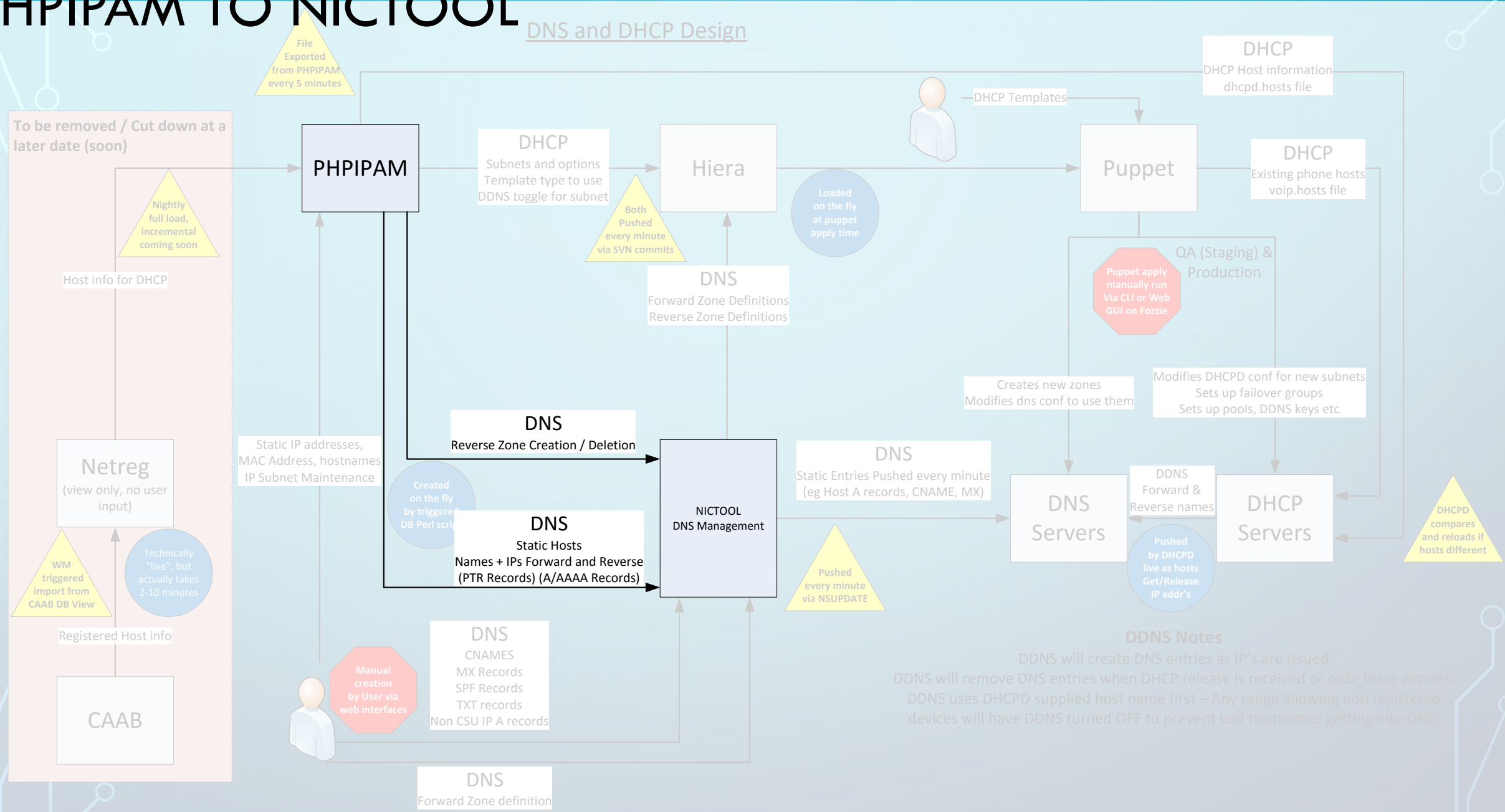
PHPIPAM - DHCPD HOSTS



PHPIPAM – DHCPD HOSTS

- PHPIPAM keeps track of all registered PC's, with a specific type assigned to each (Corporate Device, Printer, AV etc)
- A DHCPD hosts file is produced with each device assigned into host entries, with DDNS hostnames and domains assigned
- Fixed addresses are also included for those hosts that have been allocated static IP's within IPAM
- The DHCPD hosts file is produced by PHIPAM, then pulled by the DHCP servers every 5 minutes. If a difference is found, the new file is copied into the working configuration and the service reloaded.

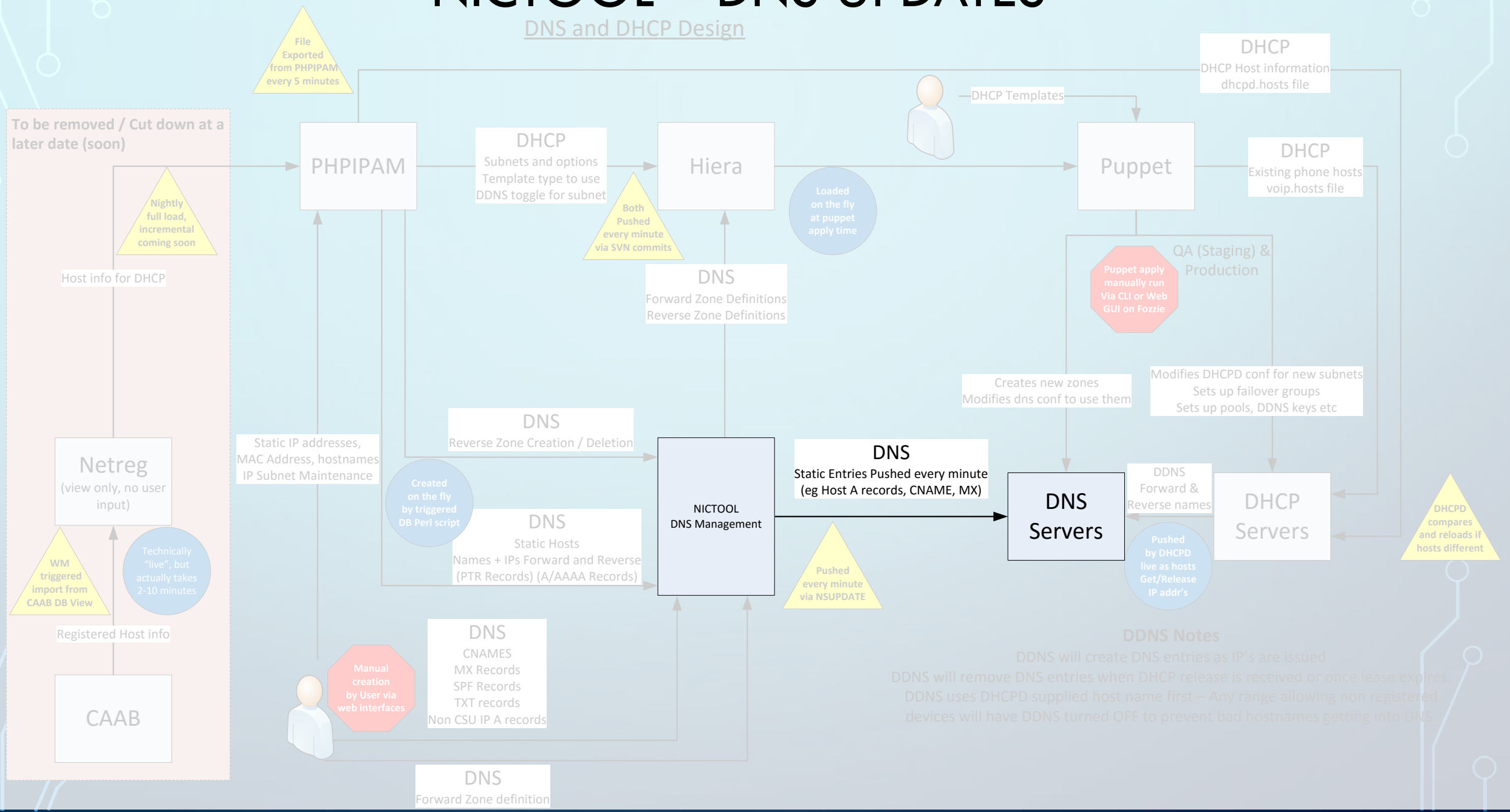
PHPIPAM TO NICTOOL



PHPIPAM TO NICTOOL

- The biggest integration script inside the system
- Written in Perl
- Triggered by DB actions inside PHPIPAM, with instantaneous updates
- Handles creation, deletion and updates for reverse zones in NicTool
- Also handles conversion from Decimal IP's to network subnets, logic around which subnet sizes to trigger reverse zone creation on
- Handles PTR, A and AAAA records for host records
- Talks to NicTool via the standard API
- Saves the NicTool ID's back to PHPIPAM for deletes / modifications

NICTOOL – DNS UPDATES



NICTOOL – DNS UPDATE METHOD

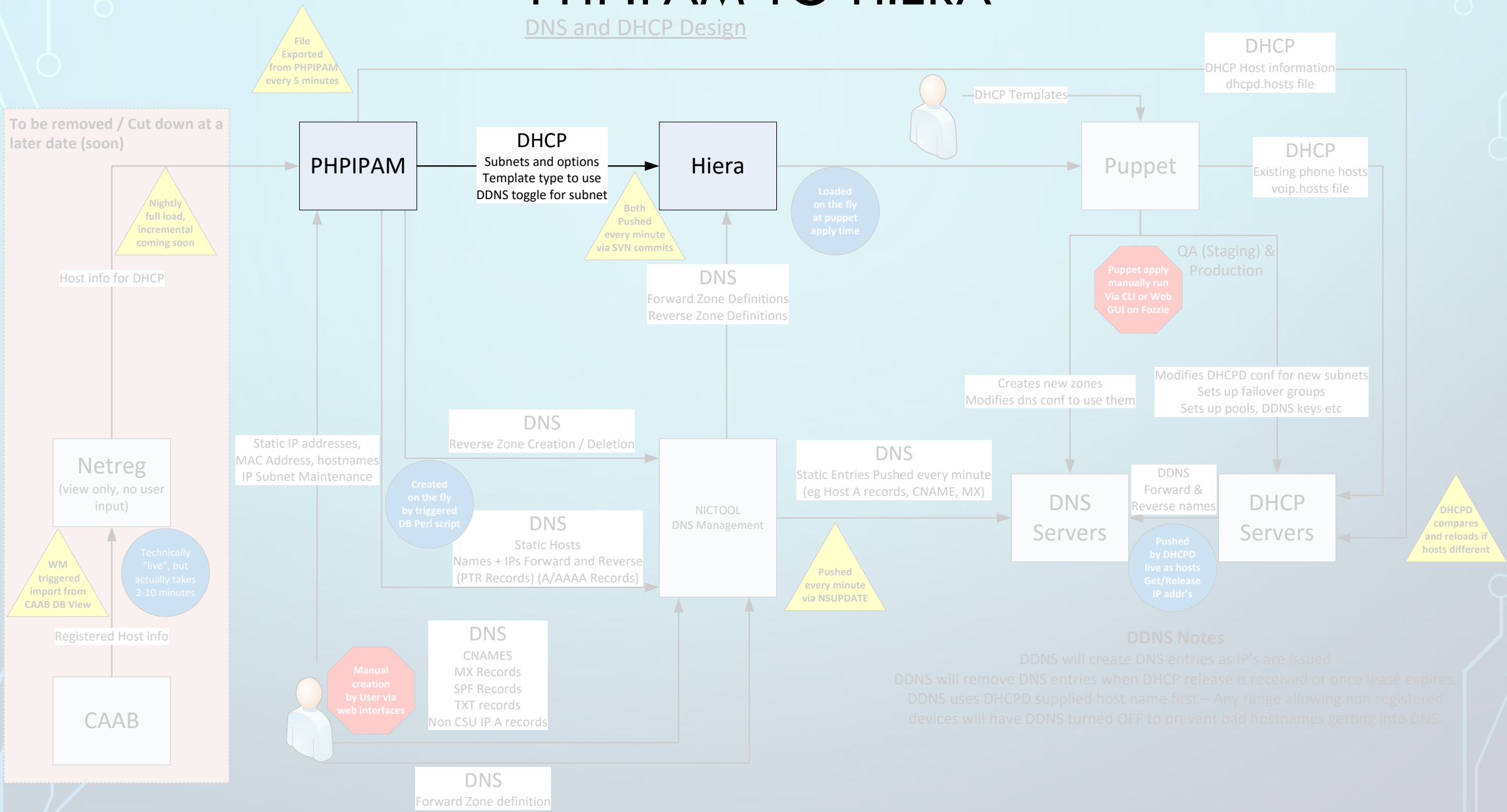
- NicTool didn't have a method to update BIND that worked with DDNS updates from DHCP
- An nsupdate style update method was written and submitted back to the author
- It is now a standard feature inside NicTool for anybody that requires it

NICTOOL – DNS UPDATES

- NicTool does an export each minute for DNS changes
- Any new changes that are found are exported into a nsupdate log
- The system runs the nsupdate log into BIND once the export is complete
- If no errors are returned, the export is marked as successful, and the timeframe for pending updates is reset
- If an error is encountered, NicTool will continue to try to do the update every minute
- 99% of issues we have seen relate to new zones that have not been created by puppet yet and are just awaiting a puppet apply to go live

PHPIPAM TO HIERA

DNS and DHCP Design

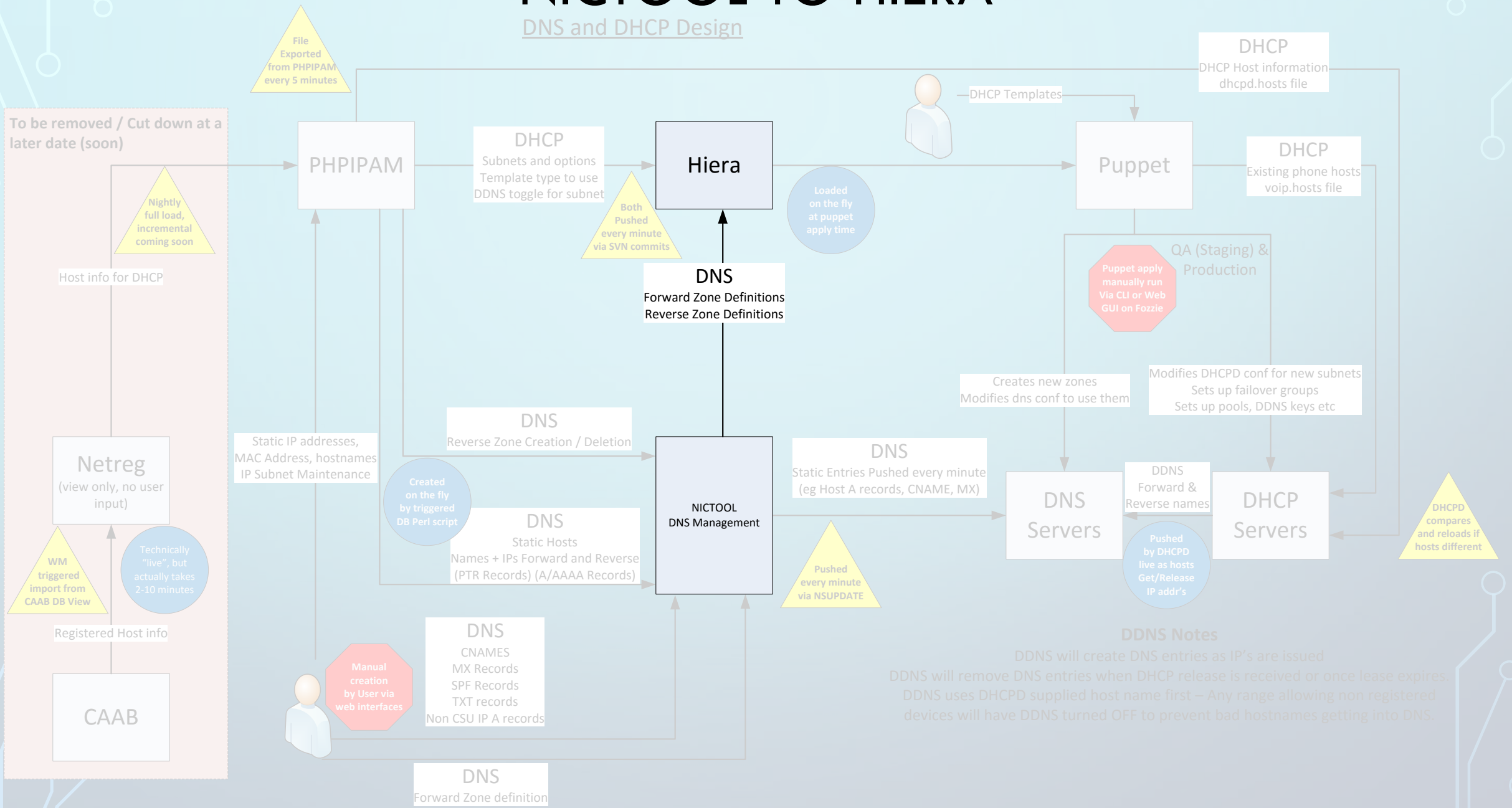


PHPIPAM TO HIERA – EXPORT DHCPD ZONES

- PHPIPAM Zones are exported to Hiera in standard YAML markup
- Both IPv4 and IPv6 zones are exported
- Ranges are exported based on /24 subnets or /64 subnets, allowing larger subnets to exist while still providing for standard DNS zone sizes
- Zones are used to provide for DDNS entries in DHCPD
- Exported every minute, compared to the previous file and synchronised to Hiera via SVN if a difference is found

NICTOOL TO HIERA

DNS and DHCP Design



NICTOOL TO HIERA – EXPORT NAMED ZONES

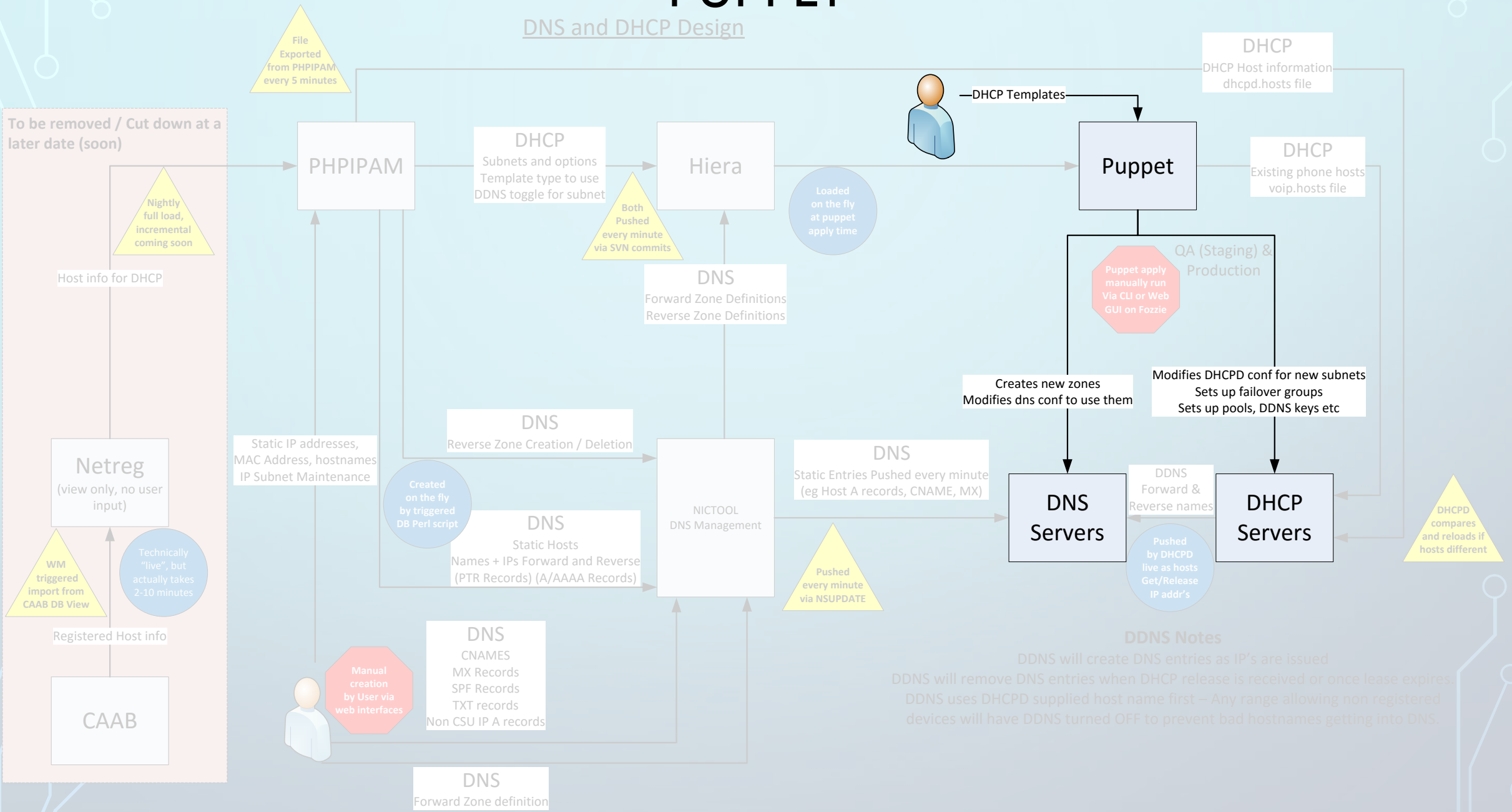
- The DNS side of the previous export
- Handled in the same export as the PHPIPAM data
- Exports YAML formatted DNS forward and reverse zones

```
bind::for_zones:
  bfdc.com.au: {}
  cappe.edu.au: {}
  charlessturt.ca: {}
  charlessturt.university: {}
  charlessturtuniversity.ca: {}
  csu.edu.au: {}
```

```
bind::rev_zones:
  0.0.0.0.0.0.0.0.0.0.0.0.0.0.d.2.5.0.4.2.ip6.arpa: {}
  0.0.0.0.0.0.0.0.1.0.0.0.0.0.d.2.5.0.4.2.ip6.arpa: {}
  0.0.0.0.0.0.0.1.1.0.0.0.0.0.d.2.5.0.4.2.ip6.arpa: {}
  0.0.0.0.0.0.0.7.0.0.0.0.0.0.d.2.5.0.4.2.ip6.arpa: {}
  0.0.0.0.0.0.0.8.0.0.0.0.0.0.d.2.5.0.4.2.ip6.arpa: {}
 10.2.10.in-addr.arpa: {}
  10.200.10.in-addr.arpa: {}
  100.166.137.in-addr.arpa: {}
  101.166.137.in-addr.arpa: {}
```

PUPPET

DNS and DHCP Design



PUPPET

- Puppet controls the configuration files for both BIND and DHCPD
- Uses the Hiera files to decide which zones to create, which configurations to build and how to do so
- Puppet apply must be run manually at this time to avoid accidental modification of production services

PUPPET - BIND

- Puppet uses the Hiera file produced earlier to create the BIND configuration file
- Zones that did not exist before are created with a standard skeleton zone with just the blank zone and name servers
- Zones that are removed do not have their zone files overwritten to ensure a zone is not removed inadvertently and configuration lost
- DDNS is configured for every zone using a DNSSEC key, allowing both NicTool and DHCPD to push DNS updates through as required

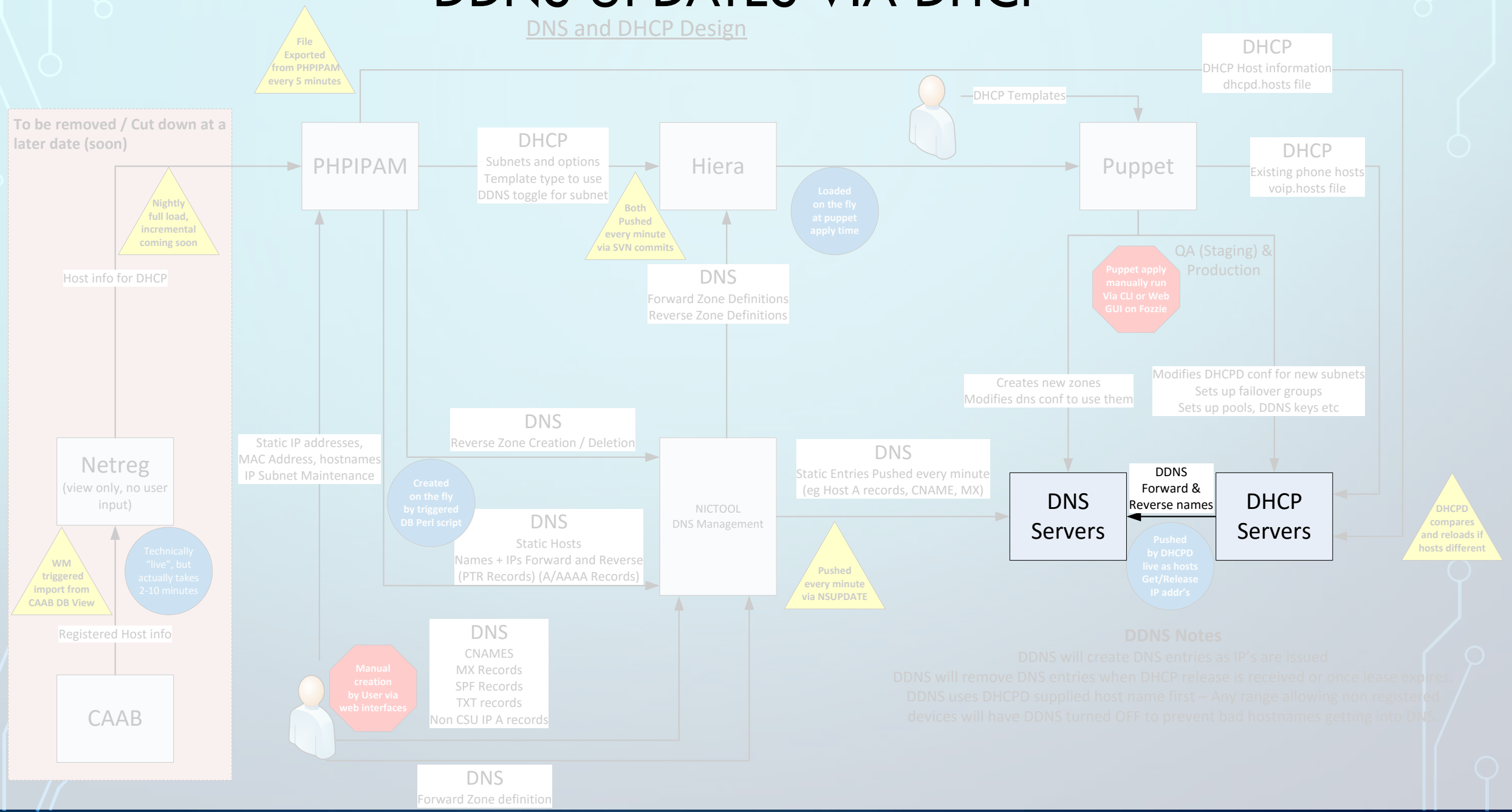
PUPPET - DHCPD

- DHCPD is configured using the Hiera file for each DHCPD server
- Master and slave are set within Node files for each server
- Puppet builds an appropriate configuration for the master and slave nodes to work together
- Includes Failover settings in the main configuration, and inside each subnet as needed

PUPPET - DHCPD

- Each Subnet is created according to the specified template, allowing different subnets to have different rules set through PHPIPAM
- DDNS settings are applied to each subnet based on the “DHCP updates” flag, allowing DHCP DNS updates to be turned on or off as required per subnet

DDNS UPDATES VIA DHCP



DYNAMIC DNS UPDATES VIA DHCP

- DHCPD is configured to provide dynamic updates to DNS as hosts receive DHCP leases
- DDNS entries are timed to line up with the lease time, as long as you keep renewing, you'll keep the DNS entry
- Once a lease expires, the DNS entry is removed
- Works for both dynamic leases and fixed addresses leases, as long as the host uses DHCP to get the IP

TROUBLE WE HAVE HAD SO FAR

- Windows servers do not support DNSSEC keys natively
- Complicated AD key issuing system required, not yet implmented
- MS domain addresses use “non standard” naming scheme which BIND has disabled by default

TROUBLE WE HAVE HAD SO FAR

- PHPIPAM does not paginate devices
- We have around 20,000 devices in there now, including staff PC's, WAP's, switches, servers, AV equipment, Facilities management devices etc
- The devices page will not load with IE, Chrome struggles, Firefox works but takes a minute or two before its useable
- The next major version of PHPIPAM (v1.3) will have fully re-written device management to address this problem

QUESTIONS?

ADDENDUM

- Some extra things you might want to see if you are viewing this presentation later

DHCPD ZONE EXPORT EXAMPLE – IPV4 AND IPV6

```
dhcpd::subnets:
137.166.62.0:
    template: residences
    dhcp_update: false
    zones:
      - 62.166.137.in-addr.arpa
      - 63.166.137.in-addr.arpa
    mask: 255.255.254.0
    range:
      - 137.166.62
      - 137.166.63
```

```
dhcpd6::subnets:
  2405:2d00:200:1100::/64:
    template: server
    dhcp_update: false
    zones:
      - 0.0.1.1.0.0.2.0.0.0.d.2.5.0.4.2.ip6.arpa
    range:
      - 2405:2d00:200:1100
```