Australian Synchrotron
Turning bright ideas into brilliant outcomes

Using Linux containers for EPICS gateways

EPICS User Meeting - Melbourne 2015

Andreas Moll

Andrew Starritt

Supported by

Australian Government

State Government Victoria

OUTLINE

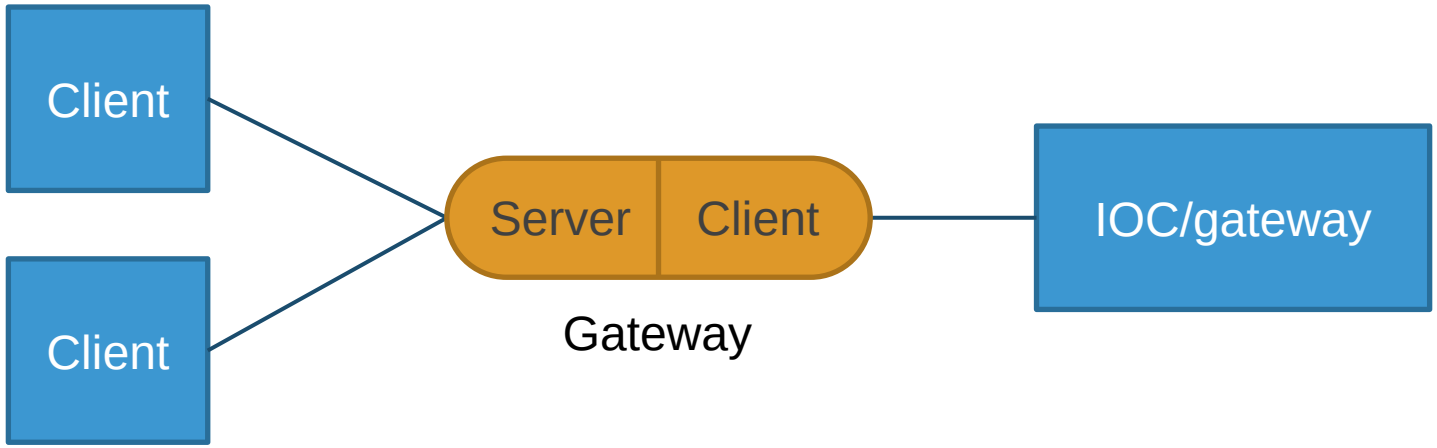
- EPICS gateways
- History of EPICS gateways at the Australian Synchrotron (AS)
- The current architecture
- Linux Containers
- Provisioning with Fabric
- Performance
- Summary

EPICS GATEWAYS

EPICS Gateway is Channel Access Server and Channel Access Client

Main features

- Performance - protect server from large number of client connections
- Security - additional access security
- Network - bridge subnets



Gateways publish their own PVs

GATEWAYS AT THE AUSTRALIAN SYNCHROTRON

Read-only access to all PVs published at the AS. For example:

- allows beamlines to read accelerator PVs (e.g. beam current)
- allows monitoring the facility from your office desk

HISTORY

The physical age

- Physical machines running multiple gateways
- Accelerator network as **backbone**
- No access for beamlines to other beamlines' PVs

The virtual age

- **Four virtual** EPICS gateway machines (CentOS 5)
- Each virtual machine hosts a number of gateway instances
- All gateway machines are connected by a **common virtual LAN** (GIN – Gateway Interface Network)
- Forward (requests from GIN) and Reverse (requests from subnet) on different virtual machines
- Each gateway client side is limited to a whitelist of all gateway servers except itself
- Virtual network interfaces for the gateways
- IOC for virtual machine status

HISTORY

The problem

- Had to add more gateways to cover additional VLANs (**19** gateways in total)
- Maximum number of virtual network cards: **10**
- Distribute gateways over more virtual machines

  ➡ Complex structure with many VMs running a number of gateways

New architecture: Group forward and reverse gateways together

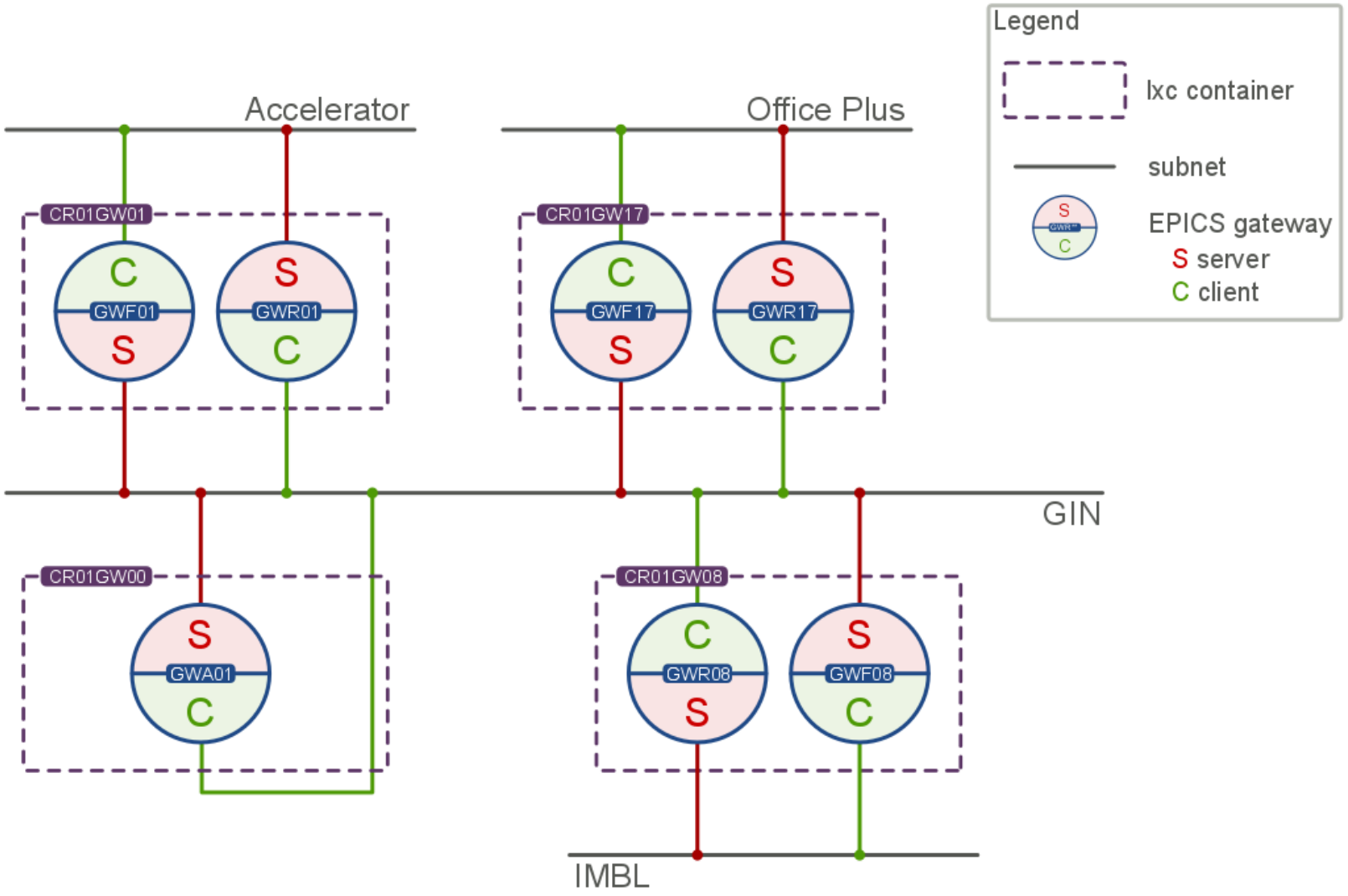
Solution #1: a separate VM for each pair of gateways (forward/reverse)

- ➕ easy to maintain and operate
- ➖ huge overhead from each VM (especially RAM consumption an issue)

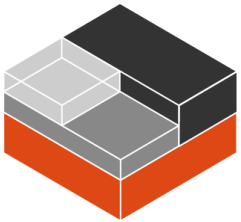
Solution #2: wrap pair of gateways in Linux containers running on a single VM

- ➕ even easier to maintain and operate
- ➕ little overhead from containers
- ➖ new technology

NEW ARCHITECTURE



LINUX CONTAINERS (LXC)



Operating-system-level virtualisation

- virtual environment (own processes and network space)
- **not** a full-fledged virtual machine

➡ allows running multiple isolated Linux systems (containers) on a single Linux control host

lxc = cgroups (limits and isolates resource usage) + isolated namespaces

- Requires kernel 2.6.24 or higher
- Fully supported in latest Linux distributions (e.g. CentOS 7)

See: https://linuxcontainers.org/

TECHNICAL IMPLEMENTATION

- Uses the VM stack at the Australian Synchrotron (see poster **WEM303**)
- **CentOS 7.1** host with CentOS 6.6 Linux containers
- Virtual machine with 2 CPUs, 4 GB RAM
- Virtual machine has access to **all VLANs**
- Network cards are implemented by the containers (no <10 limit)
- Two-way gateways: PV list access rules (**avoid loops**)
  - Forward gateway (PVs available to GIN):
    Deny requests from same gateway GIN client interface
  - Reverse gateway (PVs available to subnets):
    Deny requests from same gateway subnet client interface
- **Single-way gateway** (GIN-GIN) in order to monitor other gateways and make host information available as PVs (special IOC running in each container)

FABRIC

Management of Linux containers and gateways with a **Fabric script**

Fabric is a Python library and command-line tool for streamlining the use of SSH for application deployment or systems administration tasks

see: http://www.fabfile.org/

Provides:

- **local** and **remote** shell commands (extremely easy to run commands over ssh)
- support for **multiple hosts** (for ssh)
- **sudo** option
- uploading/downloading files
- **prompts** for user input
- auto-generation of command **help** from Python docstrings

FABRIC FOR GATEWAYS

Gateway settings stored in **configuration file** (JSON)

```
{
“template_container”: “CR01GW_template”,
“username”: “ics”,
“hostname_template”: “CR01GW%02d”,
“containers”: [
 {
  “hostname”: “CR01GW01”,
  “enabled”: true,
  “label”: “Accelerator <-> GIN”,
  “interfaces”: [
   {
    “name”: “eth0”,
    “vlan”: “vlan901”
```

New gateways created via **Fabric script** and **template gateway** Linux container

- very quick creation of additional gateways (only define gateway in config file)

Fabric script allows to manage a **single gateway** or **all gateways** at once

- update of software is only one command

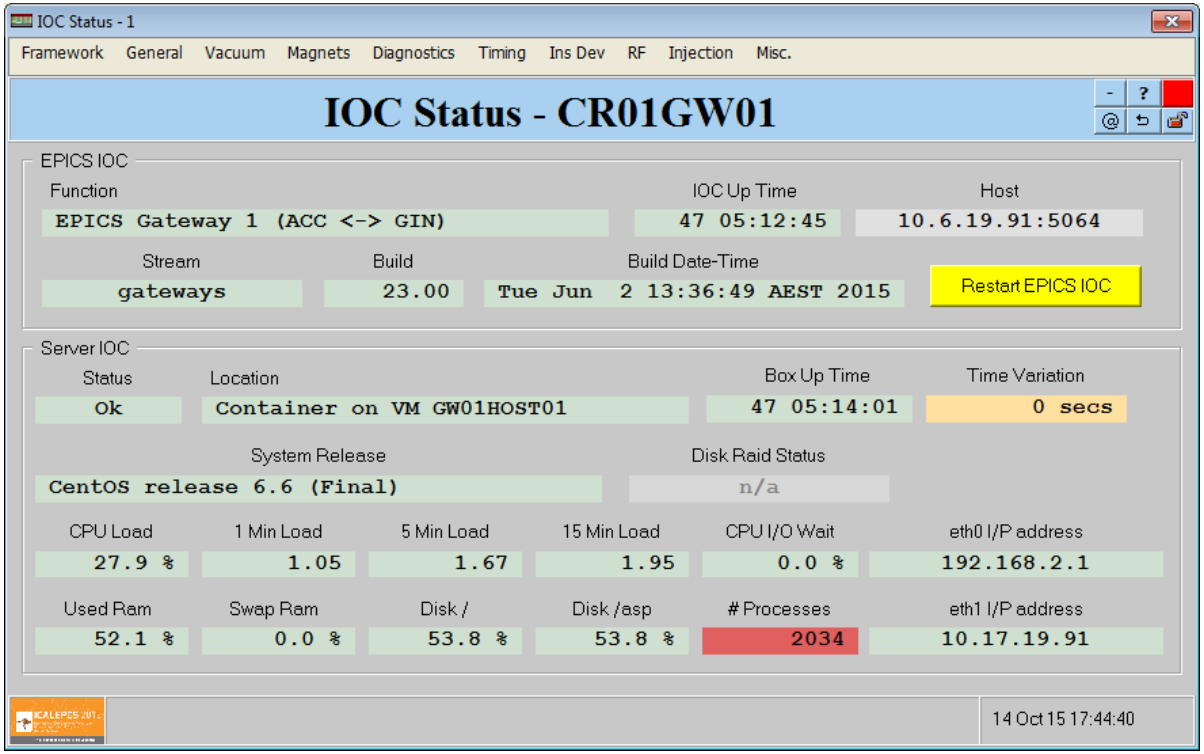
FABRIC FOR GATEWAYS

```
You can use the following commands to manage the EPICS gateway containers:

 gw_manage --list              Lists all available commands to manage containers
 gw_manage status              Shows the current status of all containers
 gw_manage status:[gateway]    Shows the detailed status information for the
                               gateway with the name [gateway] or number
 gw_manage status_gateways     Displays the status of the EPICS gateways
 gw_manage start_gateways      Starts the EPICS gateways
 gw_manage stop_gateways       Stops the EPICS gateways

[                ~]# gw_manage --list
Available commands:

    create             Creates all containers or the specified container.
    delete             Deletes all containers or the specified container.
    deploy             Deploys the EPICS gateways to all containers or the specified container.
    print_wiki_table   Prints the Wiki table structure for the specified container or all containers.
    set_build          Sets the active build for all containers or the specified container.
    shutdown           Graceful shutdown of all containers that are running or the specified container.
    start              Starts all containers that are not yet running or the specified container.
    start_gateways     Starts the EPICS gateways for all containers or for the specified container.
    start_services     Starts the EPICS services for all containers or for the specified container.
    status             Displays the status for all containers or for the specified container.
    status_gateways    Displays the status of the EPICS gateways for all containers or for the specified container.
    status_services    Displays the status of the EPICS services for all containers or for the specified container.
    stop               Stops all containers that are running or the specified container.
    stop_gateways      Stops the EPICS gateways for all containers or for the specified container.
    stop_services      Stops the EPICS services for all containers or for the specified container.
    update_hostfiles   Updates the host files of all containers or the specified container.
[                ~]#
```

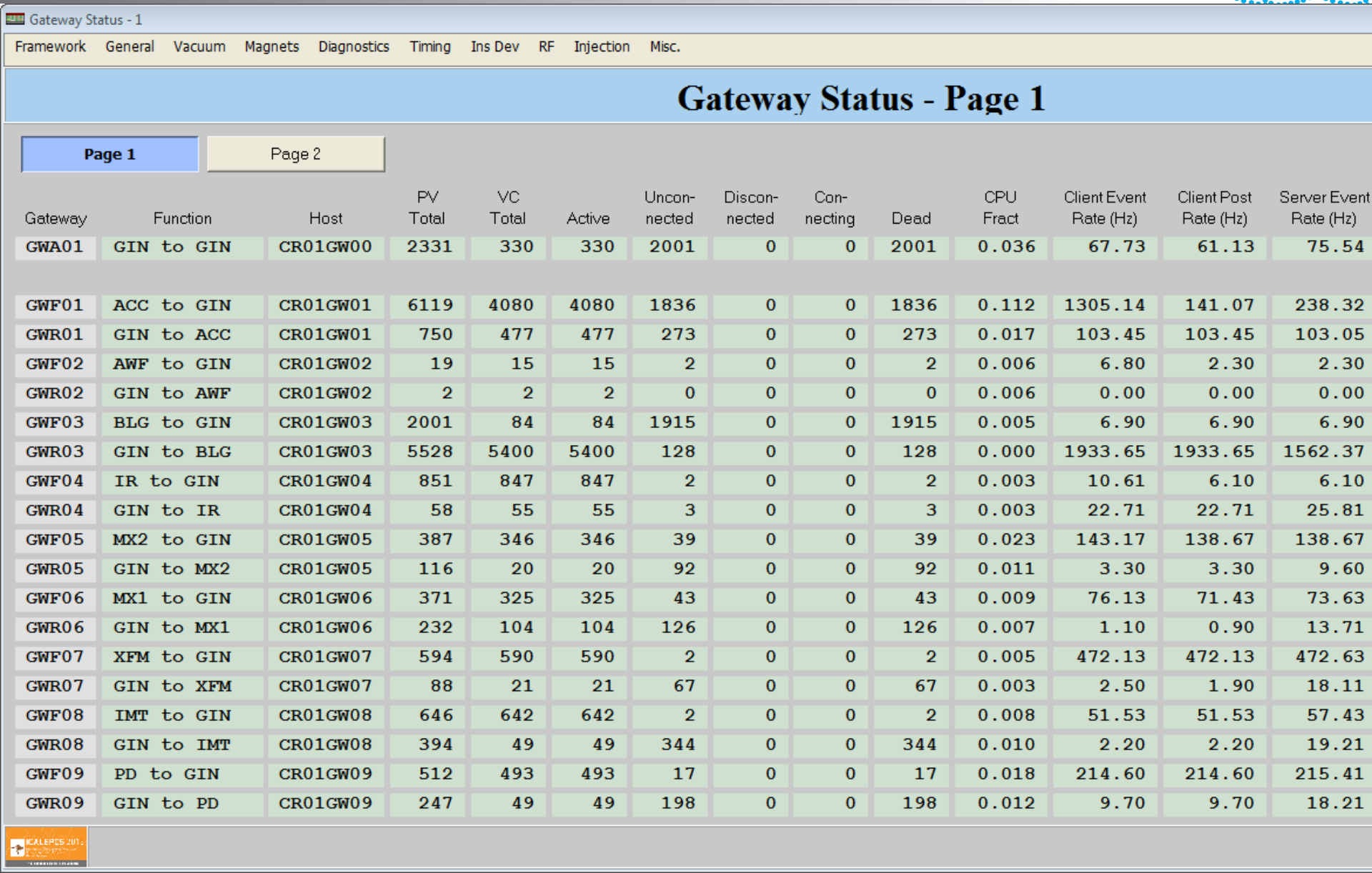
PERFORMANCE

New gateway system in production since **June**

- No problems
- Low CPU load: on average < 40%
- Low memory consumption: ~50%

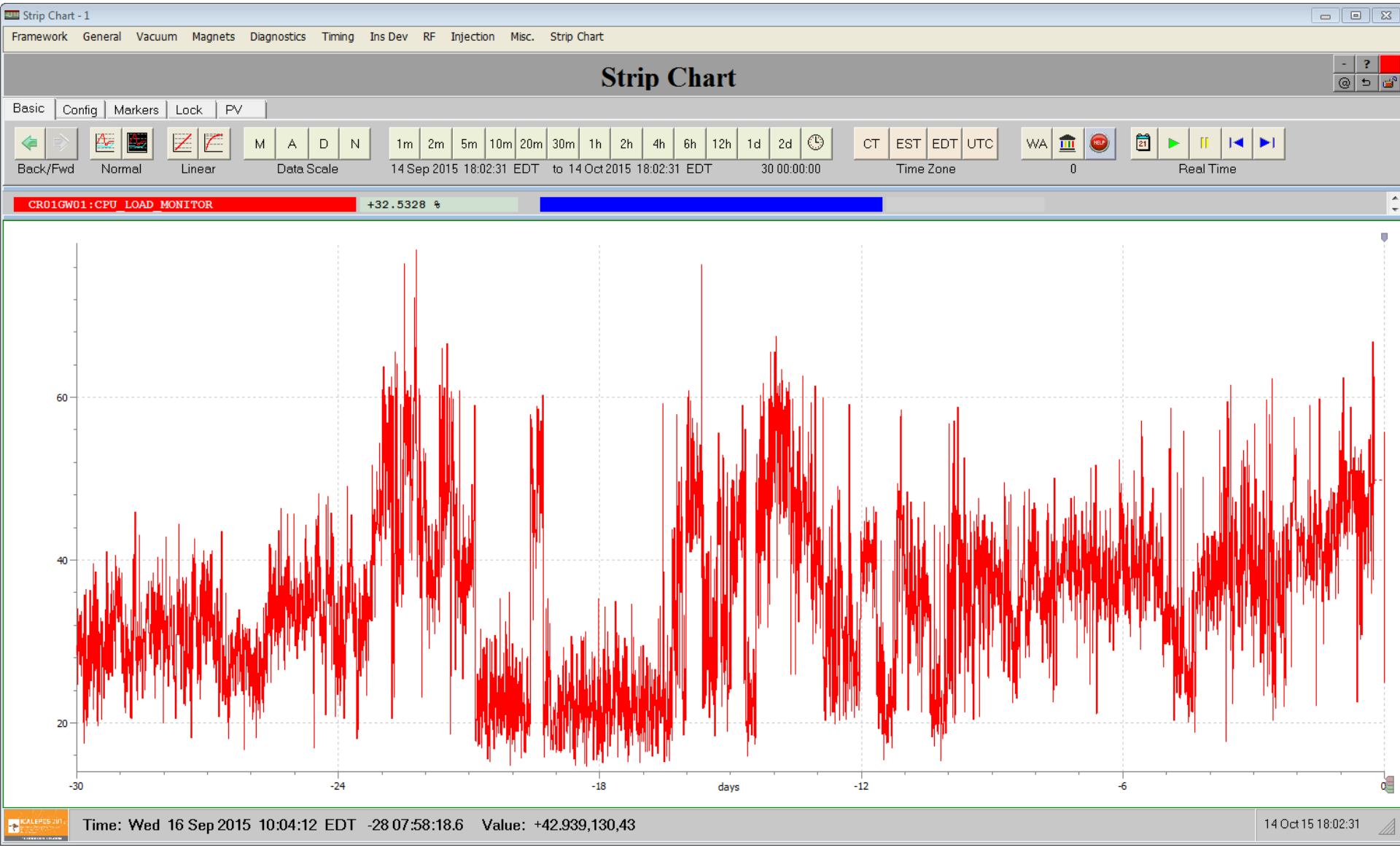


PERFORMANCE



Gateway	Function	Host	PV Total	VC Total	Active	Uncon-nected	Discon-nected	Con-necting	Dead	CPU Fract	Client Event Rate (Hz)	Client Post Rate (Hz)	Server Event Rate (Hz)
GWA01	GIN to GIN	CR01GW00	2331	330	330	2001	0	0	2001	0.036	67.73	61.13	75.54
GWF01	ACC to GIN	CR01GW01	6119	4080	4080	1836	0	0	1836	0.112	1305.14	141.07	238.32
GWR01	GIN to ACC	CR01GW01	750	477	477	273	0	0	273	0.017	103.45	103.45	103.05
GWF02	AWF to GIN	CR01GW02	19	15	15	2	0	0	2	0.006	6.80	2.30	2.30
GWR02	GIN to AWF	CR01GW02	2	2	2	0	0	0	0	0.006	0.00	0.00	0.00
GWF03	BLG to GIN	CR01GW03	2001	84	84	1915	0	0	1915	0.005	6.90	6.90	6.90
GWR03	GIN to BLG	CR01GW03	5528	5400	5400	128	0	0	128	0.000	1933.65	1933.65	1562.37
GWF04	IR to GIN	CR01GW04	851	847	847	2	0	0	2	0.003	10.61	6.10	6.10
GWR04	GIN to IR	CR01GW04	58	55	55	3	0	0	3	0.003	22.71	22.71	25.81
GWF05	MX2 to GIN	CR01GW05	387	346	346	39	0	0	39	0.023	143.17	138.67	138.67
GWR05	GIN to MX2	CR01GW05	116	20	20	92	0	0	92	0.011	3.30	3.30	9.60
GWF06	MX1 to GIN	CR01GW06	371	325	325	43	0	0	43	0.009	76.13	71.43	73.63
GWR06	GIN to MX1	CR01GW06	232	104	104	126	0	0	126	0.007	1.10	0.90	13.71
GWF07	XFM to GIN	CR01GW07	594	590	590	2	0	0	2	0.005	472.13	472.13	472.63
GWR07	GIN to XFM	CR01GW07	88	21	21	67	0	0	67	0.003	2.50	1.90	18.11
GWF08	IMT to GIN	CR01GW08	646	642	642	2	0	0	2	0.008	51.53	51.53	57.43
GWR08	GIN to IMT	CR01GW08	394	49	49	344	0	0	344	0.010	2.20	2.20	19.21
GWF09	PD to GIN	CR01GW09	512	493	493	17	0	0	17	0.018	214.60	214.60	215.41
GWR09	GIN to PD	CR01GW09	247	49	49	198	0	0	198	0.012	9.70	9.70	18.21

PERFORMANCE



SUMMARY

- New gateway architecture at the Australian Synchrotron
- Uses Linux containers on a CentOS 7 virtual machine
- Fabric as provisioning tool
- Very stable setup with low memory and CPU load footprint
- Easy to extend with additional gateways

Backup

FIRST VIRTUAL GATEWAY DESIGN