Dynfarm: A Dynamic Site Extension

IOP Conf. Series: Journal of Physics: Conf. Series **898** (2017) 082017 doi:10.1088/1742-6596/898/8/082017

Dynfarm: A Dynamic Site Extension

V. Ciaschini[1], **D. De Girolamo**[2]

INFN CNAF, Viale Berti Pichat 6/2, 40126, Bologna, Italy

E-mail: [1]vincenzo.ciaschini@cnaf.infn.it

E-mail: [2]donato.degirolamo@cnaf.infn.it

Abstract. Requests for computing resources from LHC experiments are constantly mounting, and so are their peak usage. Since dimensioning a site to handle the peak usage times is impractical due to constraints on resources that many publicly-owned computing centres have, opportunistic usage of resources from external, even commercial, cloud providers is becoming more and more interesting, and is even the subject of upcoming initiative from the EU commission, named HelixNebula.

While extra resources are always a good thing, to fully take advantage of them they must be integrated in the site's own infrastructure and made available to users as if they were local resources.

At the CNAF INFN Tier-1 we have developed a framework, called dynfarm, capable of taking external resources and, placing minimal and easily satisfied requirements upon them, fully integrate them into a pre-existing infrastructure and treat them as if they were local, fully-owned resources.

In this article we for the first time will a give a full, complete description of the framework's architecture along with all of its capabilities, to describe exactly what is possible with it and what are its requirements.

1. Introduction

Commercial clouds are offering resources to scientific computer centres for heavily reduced prices or even for free. These resources are extremely useful to provide additional computing power to cope with the ever-increasing requests from experiments, but are also poorly integrated, if at all, with the rest of the centre's ones, and therefore typically require both experiment awareness and additional manpower to be fully exploited.

Dynfarm tries to remove both these requirements by taking charge of the external resources and making them indistinguishable (except for network speed and latency) from internal ones, thus removing most disadvantages.

These article will first detail the architecture of the system, and subsequently describe in general terms two different setups. Detailed descriptions of these last two may be found here [1], here [2] and here [3].

2. Architecture

The main idea behind dynfarm is that the main obstacle to full integration is that the remote machines are not part of whatever job submission system is available on the site, and the major difficulty to do this is usually the fact that the two networks (the site's and the remote resources') are unconnected except by the general Internet.

Dynfarm takes care of these difficulties and additionally provides tools to manage all remote hosts at the same time.

A bird's eye view of the architecture is shown in Figure 1.

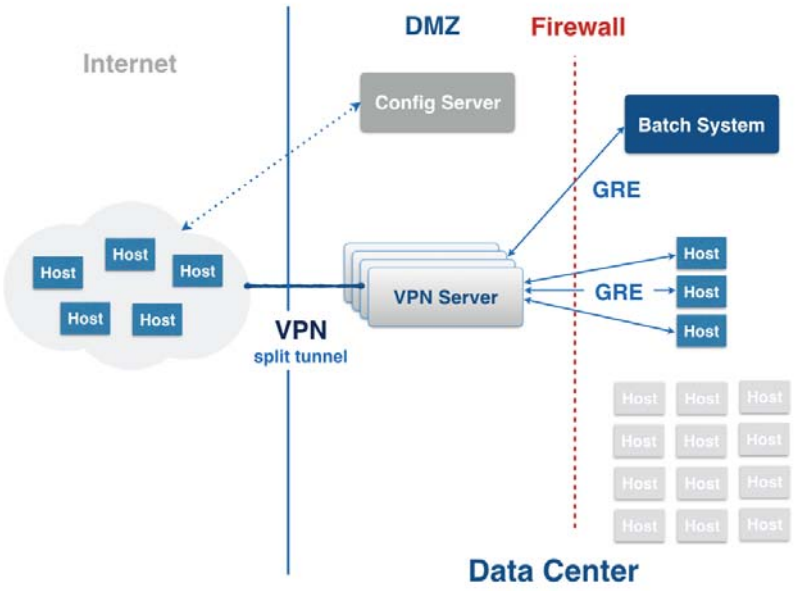

Figure 1. Dynfarm architecture

2.1. Requirements

Requirements are simple. The remote hosts must have outgoing connectivity and have to allow installation of software. Since both these requirements are also requirements for pretty much all experiment software and also allowed by all providers, there are not considered onerous. A public IP address is not mandatory. Indeed, the system has been proved to work when all remote resources are behind a NAT.

2.2. Network

In order to take advantage of remote resources, two different issues present themselves:

(i) remote resources may not be directly addressable and,

(ii) remote resources should not have full visibility of the site's resources, but only of a subset.

To satisfy the first issue, for example with sites that have resources behind a NAT system, dynfarm establishes a Host-to-LAN VPN with *Split Tunnel* between each of the remote hosts and a local (to the Data Center) VPN server. Due to the Split Tunnel, the remote hosts Internet connectivity is not generally impacted and only the traffic between the Data Center and the remote resource goes through the tunnel.

However the VPN connection itself is not enough. In fact it only allows visibility of the VPN server to the remote machine. To solve the second issue, remote hosts need to be visible to some local systems, such as local batch system, file server, local storage resources, etc... To obtain this we identified three ways:

(i) Allow the VPN network to be routed inside the site.

(ii) Put all remote machines on the same LAN/VLAN of the VPN server.

(iii) Create GRE tunnels between local hosts and the VPN server, that works as a router to permit network connectivity between local and remote systems.

Obviously the first is extremely invasive for the whole site, and the third is the most secure since it puts a strong limit on what the external resources can access.

A scheme of an example setup can be seen in Figure 2.

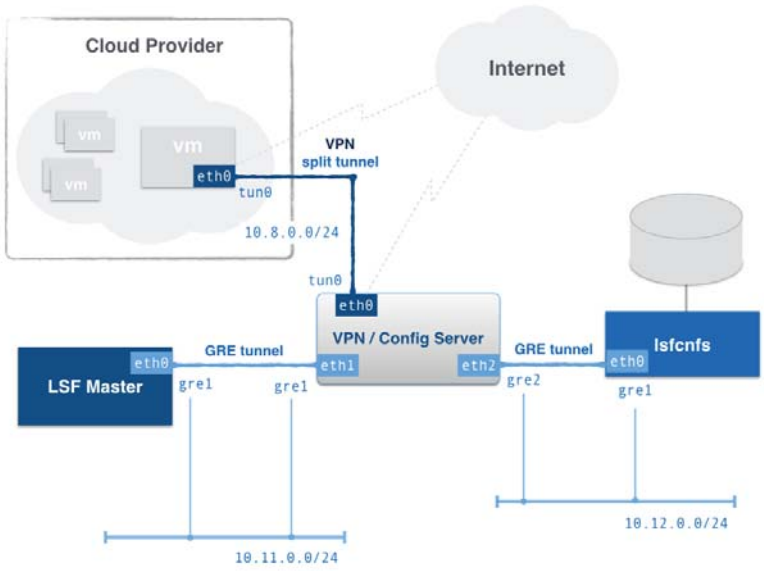

Figure 2. Dynfarm example setup

2.3. Configuration

After the remote host has been made accessible by the site, it needs to be configured. If the Job Management System requires software to be present on the slave machines, it must be installed. If environment variables or files or directories must be present, they must be created. Dynfarm has the ability to run arbitrary commands and create arbitrary files on any host that it controls, and therefore can perform any kind of customization as long as it can be scripted.

2.4. Dynfarm Composition

Dynfarm is comprised of three components:

- a server which runs on the site.
- a client which runs at startup (and only at startup) on each of the remote machines.
- a command line interface for the administrator's use.

2.5. Dynfarm Operational Workings

As the last step of the boot procedure, the client in the remote node contacts the server on the site. The server, after determining what particular provider is the source of the machines, first determines the version of the client running on the remote machine and updates it to the latest one if necessary. Afterward, it selects the appropriate VPN configuration and site customization

from its internal database and sends both to the remote host. At this point the server is no longer part of the process and it can, in fact, be shut down without affecting the functioning of any host already configured. It simply will no longer be possible to add new hosts.

When the remote hosts receive the configuration, they first create the VPN tunnel, and afterward apply the local configuration. Among the local configuration is always present a ssh key. This key is installed on the root user, and allows the administrator to run commands on the host without necessarily having a password.

At this point the system is ready and functional. However, for any management needs, the administrator has a further tool at his disposal, `remote_control`, which allows him to query a specific site to know what machines are running from there, to run commands at the same time on all of them, and to copy files from and to them.

3. Existing Setups

3.1. OpenStack

This was the first test of dynfarm with real jobs. In this setup, remote machines were running CMS jobs through on CNAF OpenStack cloud in INFN Bologna's farm, where the farm and OpenStack did not see each other. A full description of this setup can be found in [1] and [2], but in brief:

- The OpenVPN server established GRE tunnels between itself and the LSF master, the Argus Service, the Computing Element (CE) and the User Interface (UI)
- The virtual machine running on OpenStack was configure by CMS to use xrootd to access data rather than expect direct file access, along with the installation of the dynfarm client program.

The whole setup can be seen in Figure 3.

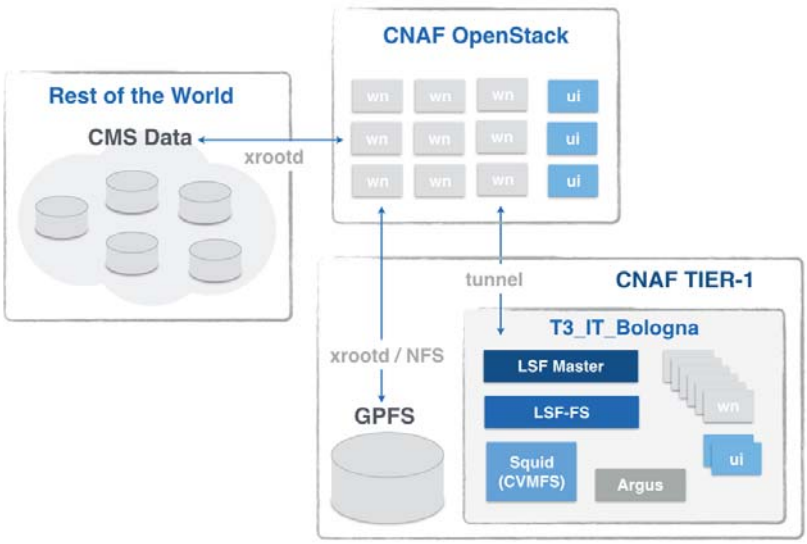

Figure 3. Dynfarm architecture for OpenStack

This setup was proved to work, and was promoted to run production CMS jobs. The performances of the setup were, thanks to the small latency to and from the other services,

almost indistinguishable from running on the local farm for jobs that did not require huge I/O (e.g. simulation) while, as expected, there was a definite drop for I/O bound jobs like analysis. The results are summarized here:

Number of jobs	Mean Efficiency
172	0.8719

3.2. Aruba

This is the second use in production of dynfarm. Aruba is one of the largest Italian cloud providers, which graciously provided us with computing power we could use free of charge for scientific computation. A full description of the setup used in this case can be found here: [3], but in brief:

- The virtual machines running on Aruba are under a NAT and cannot be distinguished from each other from the outside.
- A AFM GPFS cache was installed on Aruba to allow performing access to the shared filesystem required by GPFS.
- A squid cache was installed on Aruba to provide speedy access to experiment data.

The whole setup can be seen in Figure 4:

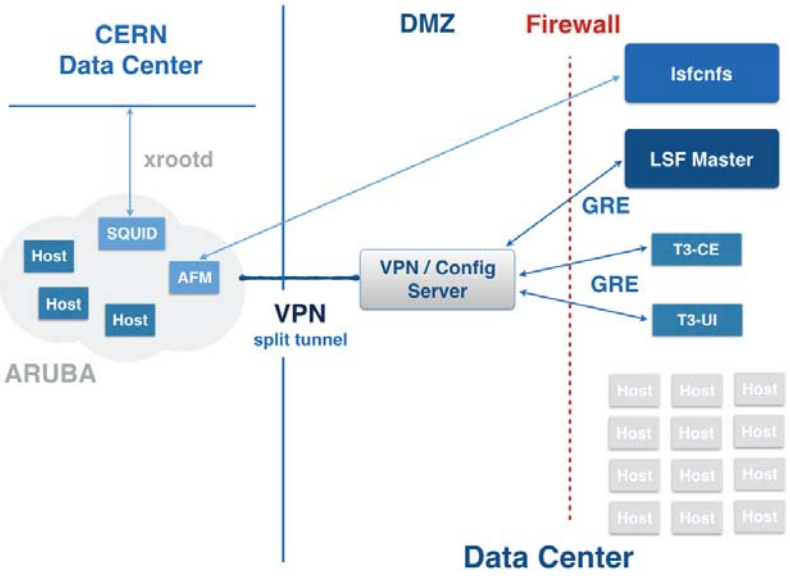

Figure 4. Aruba architecture

This setup is used in production and has proved to be stable. The performance, while numerically different due to the greater latency, is still similar and is detailed in Figure 5.

3.3. Conclusions

The performance drop that inevitably happens when data passes through the general Internet rather than being accessed through local storage has been proven very significant for some workloads, but absolutely minimal for others.

While dynfarm is not a silver bullet it has proven to be a viable solution to offload some of the site's workloads to a remote provider and freeing more efficient resources for other more demanding ones.

Queue	Nodetype	Njobs	Avg_eff	Max_eff	Avg_wct	Avg_cpt
Cms_mc	AR	2984	0.602	0.912	199.805	130.482
Cms_mc	T1	41412	0.707	0.926	117.296	93.203

Figure 5. Results of dynfarm usage on Aruba

References

[1] G. Codispoti, R. di Maria, C. Aiftimiei, D. Bonacorsi, P. Calligola, V. Ciaschini, A. Constantini, D. de Girolamo, S. Dal Pra, C. Grandi, D. Michelotto, M. Panella, G. Peco, V. Sapunenko, M. Sgaravatto, S. Taneja, G. Zizzi, Elastic Extensions of Computing Centres on Clouds, PoS (ISGC 2016)023

[2] G. Codispoti, R. di Maria, C. Aiftimiei, D. Bonacorsi, P. Calligola, V. Ciaschini, A. Constantini, S. Dal Pra, D. de Girolamo, C. Grandi, D. Michelotto, M. Panella, G. Peco, V. Sapunenko, M. Sgaravatto, S. Taneja, G. Zizzi, Elastic Extension of a CMS Computing Centre Resources on External Clouds, ACAT 2016, Journal of Physics: Conference Series 762 (2016) 012013

[3] V. Ciaschini, S .Dal Pra, D. de Girolamo, L. dell'Agnello, A. Chierici, V. Sapunenko, T. Boccali, A. Italiano, Elastic CNAF DataCenter extension via opportunistic resources, PoS (ISGC 2016)031