Interoperability in Heterogeneous Resource Federations

Sebastian Wahle[1], Thomas Magedanz[2], and Konrad Campowsky[2]

[1] Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31, 10589 Berlin, Germany
sebastian.wahle@fokus.fraunhofer.de
[2] Technische Universität Berlin, Lehrstuhl Architekturen der Vermittlungsknoten (AV)
Franklinstr. 28-29, 10587 Berlin, Germany
frollen@cs.tu-berlin.de, tm@cs.tu-berlin.de

Abstract. Currently, a number of research activities worldwide focus on resource federations to enable remote resource access and the setup of large scale experiments. With the global efforts in sharing experimental facility resources across organizational boundaries, interoperability issues are becoming increasingly important. To this end, this paper describes a model that defines the necessary entities for inter resource federation scenarios. The model is then discussed by means of a concrete use case federating Panlab and PlanetLab resources. In this respect Panlab and PlanetLab are considered to be federations themselves. Our findings show that the two approaches can be matched following the general model. This allows for federating resources not only across the boundaries of administrative (organizational) domains but even across the boundaries of federations. Those federations in themselves provide collections of resources offered by several administrative domains and use different control frameworks to enable resource access and management.

Keywords: Resource Federation, Model, Panlab, PlanetLab, SFA, Teagle, Interoperability.

1 Introduction

The current trend of federation is followed by several research projects and programs worldwide. While the concept of federation can be applied to a number of fields such as identity management, networking, trust, and security, in the context of this paper we concentrate on the federation of testbeds and experimental facilities. Generally, a federation is understood to be an organization within which smaller divisions have some internal autonomy (Oxford definition). Merriam-Webster defines federal as: (1) formed by a compact between political units that surrender their individual sovereignty to a central authority but retain limited residuary powers of government; (2) of or constituting a form of government in which power is distributed between a central authority and a number of constituent territorial units.

This concept, clearly stemming from a political background, is also applied in the context of our research in sharing hardware and software resources across the borders

T. Magedanz et al. (Eds.): TridentCom 2010, LNICST 46, pp. 35–50, 2011.

of individual administrative domains. This is important because federation enables combining infrastructural network resources and services of more than one independently controlled domain which enhances the utility of testbeds significantly. This is true for the following reasons: access can be given to more resources, increasing the scale of experiments. Furthermore, individual testbeds may include unique infrastructural resources or configuration properties that allow experimenters to execute new kinds of experiments. Finally, because testbeds act as gathering points for experimenters in a given field, combining testbed resources can promote collaboration between very different communities (e.g. Internet community and Telco community) and research groups [1].

Furthermore, with the speed of network convergence and technology evolution, today's ICT systems require sophisticated heterogeneous experimental network infrastructures to design and test new solutions. The problem is that infrastructure lifetime – the time an infrastructure remains at technology's cutting edge – has decreased dramatically, making investments in expensive isolated research infrastructure more risky than they were already. This is especially true for complex cross-layer and cross-technology testbeds.

In this regard, federation can play a major role in the worldwide research activities that are currently under way to investigate alternative solutions and design the architecture of a so-called Future Internet (FI). The FI architecture discussion has been triggered by the fact the original Internet design and its protocols were devised in the Seventies and numerous fixes and extensions have been introduced since then to address a variety of problems such as scalability and security. This led to a highly heterogeneous landscape of different protocols and technologies combined with a significant increase in the overall system complexity. At the same time the Internet is still a best effort network and Quality of Service (QoS) guarantees are hard to realize. In addition to the development of innovative foundational Internet architectures, the setup and provisioning of large scale testbeds and experimental facilities is considered to be of major importance in international research in order to develop, test, and validate FI research results. Therefore, large research programs have been launched to address both the development of new Internet architectures as well as suitable experimental facilities and test environments. Examples in this context are the United States NSF programs GENI (Global Environment for Network Innovations) [2] and FIND (Future Internet Design) [3] as well as the European FIRE (Future Internet Research & Experimentation) [4],[5] initiative. The focus of GENI is on the design of experimental platforms whereas FIND is mainly addressing foundational concepts and methods for the Future Internet. In the context of GENI, there are currently five competing testbed control frameworks (TIED [6],[1], PlanetLab [7], ProtoGENI [8], ORCA [9], ORBIT [10]) under development that are organized in clusters. In the FIRE context, several projects (e.g. Onelab2 [11], Federica [14], PII [12],[13]) are contributing to the experimental facility. In Asia similar programs have been launched such as AKARI [15] in Japan. Joint Asian activities are carried out under the APAN (Asia-Pacific Advanced Network) [16] initiative, the Asia Future Internet Forum (AsiaFI) [18] as well as PlanetLab CJK (China, Japan, Korea), a joint PlanetLab cooperation by China, Japan, and Korea. An in-depth discussion and comparison between the different control framework approaches for experimental facilities has been published earlier by the authors [19].

This article proposes a generic model for federation and demonstrates an approach for federating resources between two of the above mentioned control frameworks, PlanetLab and Panlab. The implementation is partly based on the Slice Based Facility Architecture (SFA) outlined in [20]. SFA has been chosen, on the one hand because of its aim to support generic resource types and on the other hand because an implementation of it already exists for the PlanetLab federation.

The paper is organized as follows: First, an introduction to the federation model is given as well as a short comparison of the scope and some architectural aspects of the PlanetLab and Panlab control frameworks, along with an introduction to SFA. Then, a use case scenario and an implementation that satisfies the requirements of this scenario are described. Finally, we discuss the implementation along with the challenges faced and decisions taken and evaluate our findings.

It is important to note that this article focuses on issues regarding the technical design and infrastructural layout of the respective control frameworks and a possible federation between them, leaving aside organizational, business, and legal aspects.

2 Federation Model

In the introduction we gave a definition of federation which has been motivated by a political background. Such definitions are based on the concept of surrendering individual sovereignty to a central authority. This general understanding is extended in our problem field as resource federations “at eye level” are possible.

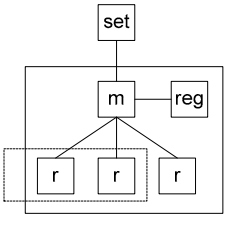

Fig. 1. Federation model entities

However, independent of the level of surrender there are similar functional entities that need to be provided to enable cross-domain and cross-technology federation concepts. The entities are shown in figure 1 and are listed below:

- Resources (r)
- Virtual grouping of resources (dotted rectangle)
- Domain managers (m)
- Registries (reg)
- Creation / setup tools (set)
- Administrative authorities (solid rectangle)

In the following we will apply different levels of "surrender" to depict several federation scenarios. The first scenario shown in figure 2 on the left hand side is what we call the "full surrender" scenario where the resources committed from domain B can be controlled via domain A. An example of the full surrender scenario is the Panlab Federation [12] where all Panlab domains allow TEAGLE, the central composition engine, to control resources inside their domain using a central registry wher resource from all participating domains are registered. Also, earlier approaches such as [21] can be modeled in this way.

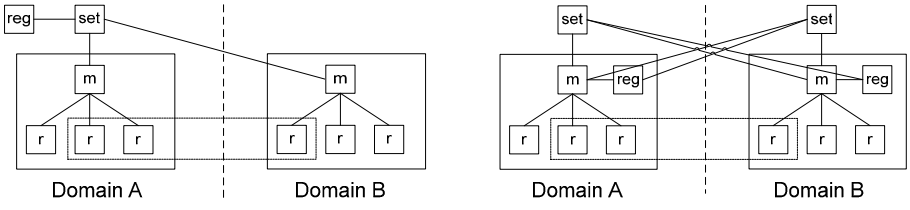

Fig. 2. Left: full surrender scenario, right: federation at eye-level scenario

The scenario shown in figure 2 on the right hand side is what we call the "federation at eye level" scenario as the participating domains allow the mutual control of resources across the borders of the individual domains. This scenario has been implemented to federate Panlab resources and resources from a private PlanetLab installation and will be explained in detail in the following sections.

Several other scenarios that are somewhere in between the both extreme scenarios explained above are possible and might be applied to satisfy various constraints and requirements in specific federation contexts. An example might be that only the registries of domains are shared allowing users to "roam" between domains and use resources from various domains without combining them across the borders of those domains.

The model has been influenced by existing federation approaches such as PlanetLab, the SFA, and Panlab.

3 PlanetLab, Panlab and SFA

This section introduces the previously mentioned frameworks and highlights the architectural concepts that are important for our further analysis and implementation of a federation scenario with mutual resource sharing between the top-level authorities.

3.1 PlanetLab Control Framework

PlanetLab is one of the platforms working under the umbrella of the GENI initiative and is being extended in this context to meet FI research infrastructure requirements. It is a global research network in the form of a distributed computing platform,

designed to support the development of new network services. The actual control framework software used by PlanetLab is called PlanetLab Central (PLC). Its primary focus lies on multiplexing the computing resources of nodes (servers) through distributed virtualization. Nodes run a minimal version of a UNIX operating system, currently the Linux flavor Fedora Core 8 (FC8) [22] and are divided into virtual containers. At present, PlanetLab is employing the Vserver [23] technology to perform this virtualization though other solutions could be used as well. The resources (reserved CPU percentage, memory and bandwidth allocations, etc.) bundled by such a virtual container are collectively called a sliver. PLC then groups these slivers into slices which are owned and subsequently administered by the party requesting the instantiation of a slice. Note that a sliver always belongs to exactly one distinct slice. Figure 3 shows this concept:

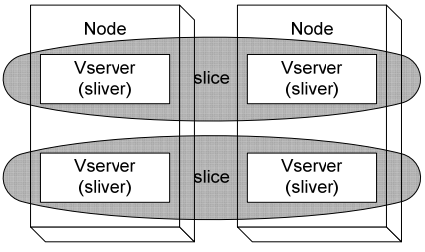

Fig. 3. Logical schematic of the PlanetLab architecture

Slices could also be described as virtual testbeds spanning parts or even the whole of the PlanetLab network of nodes. From the perspective of researchers, a slice is a *substrate-wide network of computing and communication resources capable of running an experiment or a wide-area network service* [24]. Slices also serve as the primary entity for accounting and accountability. The consuming of resources on individual nodes can be traced to a distinct slice as well as the possibly malicious behavior of experiments or other programs running within the network.

Furthermore, PLC allows the party owning a slice to grant individual researchers (users) access to it. A user associated with a slice always has access to all the slivers it comprises. Researchers can use their slices to run experiments, either by directly administering the associated slivers through SSH or by using one of several overlay technologies developed by PlanetLab or third parties [25]. Additionally, PlanetLab offers a number of operations on slices to its users, for example centralized monitoring facilities or the ability to reboot or reset all slivers in a slice.

3.2 Panlab Control Framework

One important conceptual difference between PlanetLab and the Panlab approach is that while PlanetLab's focus is centered on computing resources (nodes), Panlab aims at being a truly generic design which is able to deploy, manage, and subsequently offer arbitrary resources and services. These range from infrastructure devices like routers or VPN gateways to logical resources, for example user accounts or relations in a database. Panlab also includes computing resources as offered by PlanetLab.

The fundamental technical management authority of a Panlab domain is a Panlab Testbed Manager (PTM) which has been described in more detail in [26]. A PTM's main functionality is the execution of generic provisioning operations on resources under its control. These resources are distinctly typed, uniquely named, and can expose further specific operations in addition to PTM's generic operations. Resources follow a dynamic lifecycle; resource types can be instantiated, these instances can be worked with, (re-)configured, and finally de-provisioned and removed from the system. A high level overview of a Panlab domain and its PTM is given in figure 4.

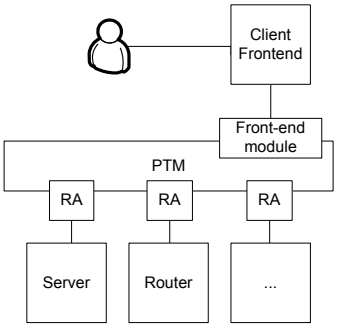

Fig. 4. Structural overview of a PTM domain

Given the high heterogeneity of resource types a PTM has to support, it cannot be directly aware of all resource specific semantics. Thus, an abstraction layer is necessary to allow for common management operations. This abstraction layer is made up of so-called resource adapters (RA) to which the PTM delegates the task of addressing resource specific types of communication. These resource adapters can be viewed as device drivers in the sense that they possess detailed knowledge about semantics of the resources instances they are responsible for. At its core level, a PTM itself is completely unaware of this nature and acknowledges resource instances merely as fundamental entities.

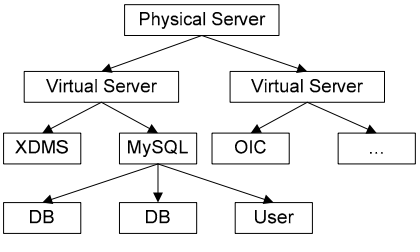

Fig. 5. Logical view of PTM's containment hierarchy

The entities are organized in a strictly hierarchical manner; a resource instance that is not at the root of the resource hierarchy has exactly one distinct parent instance and is thought of as being contained within its parent. An example of this structure is visualized in figure 5 where the arrows represent containment. The physical server at

the root of the hierarchy contains two virtual servers (Xen machines [27]) which in turn contain deployments of different software packages. In this example, these are the Fraunhofer FOKUS XML Document Management Server (XDMS) [28], an Open IMS Core (OIC) [29] and an instance of a MySQL database system [30]. The administrative issues of the MySQL instance are represented as further resources, namely individual database and database user (access credentials) instances. It must be noted, that a PTM does currently not support mechanisms for user management or access control. It assumes a trusted relationship with a client front-end to which it offers the entirety of its services through one or more front-end modules. Currently, this is the TEAGLE portal [31] which in turn provides a Virtual Customer Testbed (VCT) design tool to allow the design and provisioning of arbitrary testbed layouts. However, it is expected that additional mechanisms are needed to allow different domains to define specific access policies. This is currently under development.

3.3 Slice Based Facility Architecture

The Slice Based Facility Architecture is one of the key parts in the high level architecture of the GENI initiative. SFA defines a minimal set of interfaces and data-types to allow a federation of slice-based network substrates to interoperate. This specification was designed to allow federation among facilities like PlanetLab, Emulab [32], and other GENI frameworks. However, it is intended to support a much broader range of heterogeneous resources and services than those systems currently embody. SFA's general layout and abstractions closely resemble those of PLC. In fact, the interfaces the SFA defines represent a subset of the operations PLC exposes. Thus, PlanetLab entities can usually be directly mapped to SFA entities and vice versa, as illustrated by figure 6:

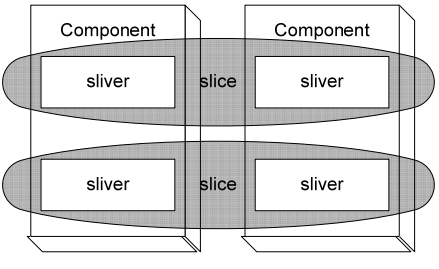

Fig. 6. Logical schematic of the SFA

The fundamental building block of the SFA is a component (CM). Components could for example correspond to physical servers, customizable routers, or access points. Their direct counterparts in PLC are nodes, which would be viewed as components in SFA. Components encapsulate a collection of resources which might include physical (for example CPU and memory allocation) as well as logical ones such as port numbers or access rights. The resources a component possesses are described in a resource specification (RSpec), the exact format of which has yet to be finalized. It is to be noted that resources don't necessarily have to be contained in a single physical device but might also be distributed across a set of devices which would

however be viewed as a single component by the SFA. A given resource always belongs to exactly one component.

The resources a component comprises can be multiplexed for use by multiple users. This can either happen via a form of virtualization, where a user is granted a virtual copy of the component's resources or by partitioning the component into several distinct resource sets. Regardless of how it is acquired, such a set of resources is called a sliver. The slivers (Vservers) created by PLC on PlanetLab nodes directly correspond to slivers in the SFA. Just like PLC does, SFA then bundles these slivers into slices. Again, a sliver always belongs to exactly one slice. Slices in SFA perform the same functions as they do in PLC, which is being the basis for accounting and accountability. Researchers who are granted access to a slice likewise gain access to all the resources represented by the slivers the slice is composed of.

Collections of components can be represented by a single aggregate. Such an aggregate can be accessed via an aggregate manager (AM) which exposes the same interface as the respective components themselves and delegates requests towards them. In the current PlanetLab implementation, central functions like the creation of slices or user management are exposed to administrators and users via a slice manager (SM), though this is not strictly defined as part of the standardized SFA interfaces. The slice manager will use its local registry to perform look-ups and is furthermore configured to know all aggregate managers of foreign parties which it will contact in order to issue provisioning requests.

4 Resource Management and Federation

In this section we discuss the proposed federation scenario ("at eye-level") and give insights into general decisions made.

4.1 Federation Scenario

The SFA describes a number of different usage scenarios, regarding different levels of federation between partners; ranging from simply providing a different user interface over shared registry services to full federation between two or more parties on equal terms. This relates to the general federation scenario shown in figure 2 on the right hand side. The scenario we are examining within the scope of this paper is that of a full federation between a private PlanetLab and a Panlab domain. Here, both parties maintain their own registry services and both parties also provide a domain manager to allow the other party to provision resources in their respective domain.

Users and administrators interact with their local domain managers (PTM and slice manager) to create and manage slices which may span both federations. The slice managers delegate look-up requests to their local registry which will in turn query the foreign registry if necessary. Provisioning requests are issued directly to the respective domain managers which forward these requests to their components. The Panlab federation mechanisms have been slightly adapted for this scenario (implementation of an aggregate manager module and lightweight SFA registry) to support the PLC and SFA mechanisms. Eventually, if the proposed federation model will be accepted and standardized APIs and interfaces will be present, this scenario will work without one side adapting to the mechanisms of the other side. However, even today the

implementation of an adaptor module making the Panlab side "PlanetLab-compatible" was feasible in terms of time investments given the modular structure of both approaches and a certain overlap in the architecture design.

4.2 Design Considerations

Our solution for realizing the above scenario takes into consideration the envisioned federation benefit and utility for both sides. Please note, that for the full federation scenario to become true, such a federation must not simply provide any level of integration and interaction, but must rather enable both parties to fully incorporate the committed resources and services into their own schema without further adjustments.

From the PlanetLab perspective, this means that PLC must be able to facilitate the same services and access rights for Panlab nodes as for PlanetLab nodes. Researchers accessing the federation via PlanetLab's slice manager must be able to request slivers from Panlab components, add them to their slices and access them in just the same way they would interact with nodes provided solely by PlanetLab itself. That implies that a PTM, or rather the resource adapter providing virtual nodes for that matter, must have a certain awareness of the requirement to deploy a PlanetLab specific flavor of a virtual node. As not all resources from the Panlab framework and its partners will be available to be shared under such circumstances, specific operational processes and agreements will be needed to manage resource access and allow for different levels of commitment for individual domains and resources. The Panlab operational framework, which is currently under development, is intended to deal with such issues by defining reference point governance processes and contracts.

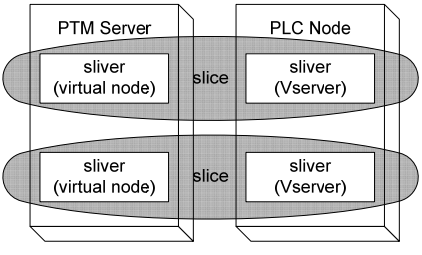

Fig. 7. PLC view on federated resources

As shown in figure 7, the SFA and through it PLC acknowledges resources provided by PTM as mere slivers and has no regard for any PTM internal layout. In the example above, these slivers represent either virtual nodes on one of PTM's physical servers or Vservers on one of PLC's nodes, both of which can equally act as slivers in a PLC slice. The federation partner that has requested these resources as well as their users can utilize them transparently, as if they were their own while being completely agnostic to the fact that they are actually provided by a different organization.

On the other hand, the PTM must also be enabled to make use of slivers provided by PlanetLab or other parties in the same way it would utilize resources under its direct control. Although SFA and PLC do not directly support PTM's notion of a

resource hierarchy, this requires little efforts. Since PLC already provides sufficient facilities (in the form of unrestricted administrative access) to allow arbitrary operations on slivers, no adjustment is needed for the PlanetLab side of the federation. PTM merely has to deploy its own management back-ends on PLC slivers, in the form of appropriate resource adapters. This already happens for PTM's own virtual nodes and the scheme can be easily adjusted to work on PLC's Vservers.

Figure 8 shows a PLC provided Vserver embedded into PTM's resource hierarchy. Note, that the example shown introduces a purely virtual *SFA* resource instance at the root of the hierarchy that bundles all slivers acquired from the SFA as children under its own hierarchy. However, this is not a technical necessity and serves only an organizational purpose.

Afterwards, the resources provided through SFA, in this example a PLC Vserver, integrate seamlessly into PTM's resource hierarchy and can be further utilized just like any other PTM resource including the possibility of deploying further subresources. Likewise, resources provisioned this way could be included into a VCT booked via the TEAGLE portal website.

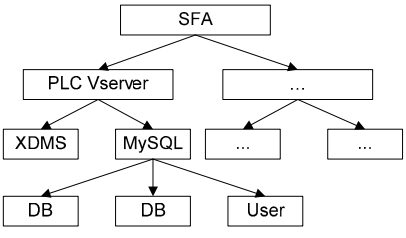

Fig. 8. PTM view on federated resources

In a further iteration, PTM's mentioned capability of creating further child resources inside a sliver acquired from the SFA could also be exposed towards other federation partners. To do so, PTM would have to advertise the sliver it was given by the SFA as another component through its aggregate manager. This virtual component would contain resources that represent the resource types PTM can deploy into the sliver. These resources could again be grouped into a (sub-)sliver and added to a slice.

However, this is not without side effects. For example one would have to define semantics for what is to happen with sub-slivers when the slivers they are contained in are being deleted. Apart from these considerations, this endeavor would currently be bound to fail for the profane reason that the current SFA implementation caches for a certain time the resources offered by an aggregate. Thus, it would be unpredictable when dynamically added resources would be picked up and made available.

5 Use Case and Prototype Implementation

This section defines a use case to be fulfilled by the implementation and gives a deeper insight into its internal architecture and layout. We will name the technologies

used and highlight key parts of the design to explain their role in the concept presented here.

5.1 Use Case Description

As a proof of concept, we have chosen to realize a simple two-way scenario that shows the implementations capabilities to provide services towards both sides of the federation:

- A researcher affiliated with PlanetLab uses his PLC credentials to access the PlanetLab slice manager. He creates (or already owns, this detail is irrelevant for this purpose) a slice and subsequently requests slivers from a PlanetLab and a Panlab node to be part of his slice.
- A researcher uses her Panlab credentials to log into the TEAGLE portal and creates a new VCT (Virtual Customer Testbed, equivalent of a slice). Using the VCT tool she adds a PlanetLab node (sliver) to her testbed and additionally chooses to deploy another software package (MySQL) onto it.

5.2 Engineering Decisions and Implementation Details

We have implemented a module, called SFAAdapter, which acts as a bridge between the internal semantics and protocols of a PTM and the SFA. In the context of PTM, this is a hybrid module in the sense that it acts as both a front-end module to PTM, serving provisioning requests issued by the SFA, and as a PTM resource adapter, making resources which federation partners offer through SFA available to other PTM internals. From the viewpoint of the SFA, this module appears just as any other federation partner, exposing the interfaces of a registry and aggregate manager.

The module itself has been implemented in the python programming language. One reason for choosing Python was that PlanetLab's implementation of the SFAexists in python. This made it possible to reuse many utility classes and functions from the existing module.

Internally, SFAAdapter comprises three main parts, shown in figure 9 as squares inside the box labeled SFAWrapper. In detail, these elements and their duties are:

- Aggregate manager (AM): As the name already implies, this entity acts as an aggregate manager as specified by the SFA. It will advertise the resources the PTM can provide when queried by the slice managers of federation partners. Furthermore, it receives provisioning requests which it will translate towards the PTM core to acquire resources.
- Resource Adapter (RA): The resource adapter part of SFAAdapter acts as the counterpart to the aggregate manager. Towards the PTM core, it poses as a native resource adapter module. It queries foreign aggregate managers about the resources they have to offer and relays the information to PTM. Subsequently, it issues provisioning requests received from the PTM side towards said aggregate managers to book resources from other parties. This module is also responsible for deploying further PTM resource adapters for the acquired resources so they can be managed by the PTM.

- Registry (Reg): This module provides a registry service as specified by the SFA. It can be queried by the PTM core as well as by remote registries and provides information about the SFA objects which this domain is responsible for. The information is obtained from a database shared between the different parts of SFAAdapter.

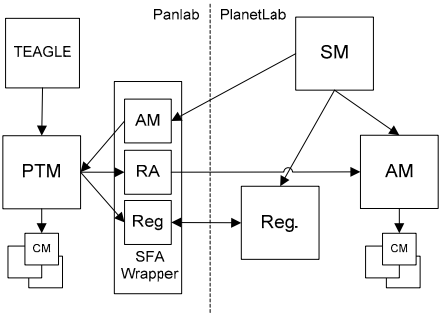

Fig. 9. Implementation layout; the arrows indicate the directions in which requests are issued

It has to be noted, that the modules providing SFA interfaces (aggregate manager and SFA registry) are both based on the original Princeton SFA implementation. This in turn means that existing SFA front-end clients can interact with them without further adjustments.

6 Proof of Concept

This section shows a glimpse of how a researcher could access federated services from each side of the federation by walking through a number of steps to set up a rudimentary testing environment. Note that in the example shown, the output is occasionally truncated for brevity and readability.

6.1 PlanetLab Perspective

A PlanetLab researcher wishes to use his PLC credentials to create a testbed environment via SFA. He has already been added to the SFA registry by a PlanetLab administrator and owns a slice (*plc.fokus.s1*) which, however, does not have any slivers associated yet:

```
#sfi.py resources plc.fokus.s1
  <RSpec ...>
    <networks>
      <NetSpec name="plc" .../>
    </networks>
  </RSpec>
```

Therefore, he first procures an overview over all available resources while at the same time saving the output for later use:

```
#sfi.py resources -o all.rspec
  <Rspec ...>
    <networks>
      <NetSpec name="ptm" ...>
        <nodes>
          <NodeSpec name=”pnode-0.ptm”>
            <net_if>
              <IfSpec addr="10.0.0.10" .../>
            </net_if>
          </NodeSpec>
        </nodes>
      </NetSpec>
    </networks>
    <networks>
      <NetSpec name="plc" ...>
        <nodes>
          <NodeSpec name=”pln0.plc”>
            <net_if>
              <IfSpec addr="10.0.0.20" .../>
          </net_if>
          </NodeSpec>
        </nodes>
      </NetSpec>
    </networks>
  </Rspec>
```

From this information, he learns that he has two nodes at his disposal, *pln0.plc* from the domain *plc* and *pnode-0.ptm* from the domain *ptm*. He adds them to his slice:

```
#sfi.py create plc.fokus.s1 all.rspec
```

The PlanetLab slice manager will now contact the PTM's aggregate manager and request it to instantiate a sliver on pnode-0.ptm. The PTM in turn contacts the appropriate resource adapter, orders it to set up a virtual node and to configure it to be PLC compatible. To give rudimentary access to researches, this especially means installing respective user credentials and configuring the sliver's SSH server. The researcher can now access the sliver and gain privileges in the same way he would access a sliver acquired from PLC:

```
#ssh -i .ssh/id_rsa focus_s1@pnode-0.ptm sudo su -
```

6.2 Panlab Perspective

The Panlab researcher mentioned in our use case opens the VCT tool via the TEAGLE portal. After entering her Panlab credentials, she starts assembling a new testbed comprising an installation of the MySQL software package on a node committed by PlanetLab (see figure 10). After carefully reviewing her setup, she issues the provisioning requests. Upon receiving these, the PTM contacts the SFA resource adapter which in turn will relay the request towards the aggregate manager of the PlanetLab side.

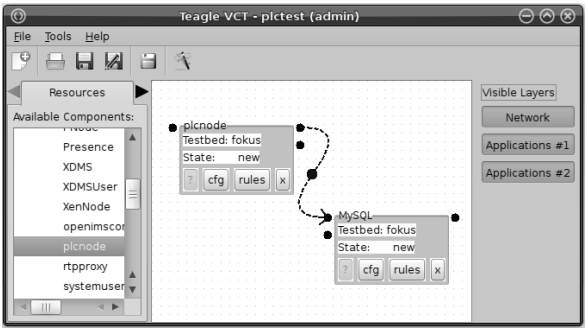

Fig. 10. VCT tool view

Since a sliver must always be part of a slice, the latter must obviously exist for the provisioning to take place. However, PTM dynamically creates those internally and hides this detail from the user. After the sliver is created by PLC, PTM accesses it and installs a number of resource adapters; among them the so called SoftwareAdapter which it will subsequently be used to deploy the requested MySQL package.

The researcher can now bring up the configuration page of the newly provisioned resource to learn some of its details. These are currently read-only values:

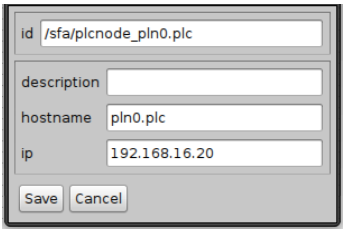

Fig. 11. Configuration page for a PLC node in TEAGLE

Note that in this example the researcher had to explicitly choose to deploy a plcnode in order to acquire a node from the PlanetLab network. In a real-life scenario a user would only choose to deploy any kind of virtual node and leave the details to the PTM internals.

7 Conclusion

It is technically possible and feasible to share resources and services between the PlanetLab and Panlab networks following the general model proposed by this paper. We had to apply an adaptor module to the Panlab Domain Manager and followed the basic SFA principles to achieve this. Both sides could benefit from the demonstrated scenario by offering a broader range of resources to their users. However, a number of things remain to be done for future efforts. As noted, the process of selecting resources is not yet completely transparent for a TEAGLE user. In the future, the

placement of virtual nodes would have to be decided fully by the PTM itself. Also, slice management could be improved. As of now, slices are not yet associated with individual VCTs as would be desirable.

Conceptually, our next steps will be to expand the implementation on the PTM side to not only commit virtual nodes but resources of arbitrary type towards the federation. This is already possible for isolated domains, however, when federating additional constraints arise. Also, the possibility of deploying further sub-resources, as already used in the TEAGLE demonstration above, could be exposed to other federation partners by dynamically advertising acquired slivers as new SFA components. Any non-technical aspects are yet completely untouched. Obviously, extensive operational agreements and procedures would have to be established before resources could be federated in the envisioned manner across several administrative domains. This includes the support for different resource access policies as well as federated identity management systems.

With this paper we showed that is possible to federated resources not only across the boundaries of administrative (organizational) domains but even across the boundaries of federations. Those federations in themselves provide collections of resources offered by several administrative domains and use different control frameworks to enable resource access and management. Although this is possible, many initiatives that are currently active in the field of resource federation would benefit from international standardization and agreed interfaces instead of bridging and adapting to each other via specific adaptor modules.

References

[1] Faber, T., Wroclawski, J.: A Federated Experiment Environment for Emulab-basedTestbeds. In: ICST (2009)
[2] National Science Foundation, GENI website, http://www.geni.net
[3] National Science Foundation, FIND website, http://www.nets-find.net
[4] European Commission, FIRE website, http://cordis.europa.eu/fp7/ict/fire
[5] Gavras, A., Karila, A., Fdida, S., May, M., Potts, M.: Future internet research and experimentation: the FIRE initiative. SIGCOMM Comput. Commun. Rev. 37(3), 89–92 (2007)
[6] Faber, T., Wroclawski, J., Lahey, K.: A DETER Federation Architecture. In: DETER Community Workshop on Cyber-Security and Test (2007)
[7] Peterson, L., Roscoe, T.: The Design Principles of PlanetLab. SIGOPS Oper. Syst. Rev. 40(1), 11–16 (2006)
[8] GENI Project Office, ProtoGENI Control Framework Overview, GENI-SE-CF-PGO-01.4 (2009)
[9] Chase, J., et al.: Beyond Virtual Data Centers: Toward an Open Resource Control Architecture. I Selected Papers from the International Conference on the Virtual Computing Initiative (ACM Digital Library) (2007)
[10] Ott, M., et al.: ORBIT Testbed Software Architecture: Supporting Experiments as a Service. In: Proceedings of IEEE Tridentcom 2005 (2005)
[11] OneLab project website, http://www.onelab.eu/

[12] Gavras, A., Hrasnica, H., Wahle, S., Lozano, D., Mischler, D., Denazis, S.: Control of Resources in Pan-European Testbed Federation. In: Towards the Future Internet - A European Research Perspective, pp. 67–78. IOS Press, Amsterdam (2009) ISBN 978-1-60750-007-0
[13] Website of Panlab and PII European projects, supported by the European Commission in its both framework programmes FP6 (2001-2006) and FP7 (2007-2013), http://www.panlab.net
[14] Campanella, M.: The FEDERICA Project - A federated infrastructure for Future Internet research. EURESCOM mess@ge, Issue 2/2008 (2008)
[15] AKARI project website, http://akari-project.nict.go.jp/eng/index2.htm
[16] Asia-Pacific Advanced Network initiative website, http://www.apan.net/
[17] Chen, M., Moon, S., Nakao, A.: Goals and Blueprint for PlanetLab CJK. Presentation at Conference for Future Internet 2008 PlanetLab BoF, Seoul, Korea, June 19 (2008)
[18] Asia Future Internet Forum website, http://www.asiafi.net/
[19] Magedanz, T., Wahle, S.: Control Framework Design for Future Internet Testbeds. e & i Elektrotechnik und Informationstechnik 126(07/08), 274–279 (2009)
[20] Peterson, L., et al.: Slice Based Facility Architecture, Princeton (2007)
[21] Fu, Y., Chase, J., Chun, B., Schwab, S., Vahdat, A.: SHARP: an architecture for secure resource peering. SIGOPS Oper. Syst. Rev. 37(5), 133–148 (2003)
[22] Fedora project website, http://www.fedoraproject.org
[23] Linux VServer project website, http://linux-vserver.org
[24] Peterson, L., Sevinc, S., Baker, S., Mack, T., Moran, R., Ahmed, F.: PlanetLab Implementation of the Slice Based Facility Architecture, Princeton (2009)
[25] PlanetLab bibliography index, http://www.planet-lab.org/biblio
[26] Wahle, S., Campowsky, K., Harjoc, B., Magedanz, T., Gavras, A.: Pan-European Testbed and Experimental Facility Federation – Architecture Refinement and Implementation. Inderscience International Journal of Communication Networks and Distributed Systems (IJCNDS), Special Issue: Recent Advances in Test-bed Driven Networking Research (forthcoming)
[27] Xen project website, http://www.xen.org
[28] XDMS project website, http://www.open-ims.org/xdms
[29] OpenIMSCore project website, http://www.openimscore.org
[30] MySQL project website, http://www.mysql.com
[31] TEAGLE portal website, http://www.fire-teagle.org
[32] Emulab website, http://www.emulab.net