

On the Design and Evolution of an Architecture for Testbed Federation

Stephen Soltesz, David Eisenstat, Marc Fiuczynski, Larry Peterson
<soltesz,deisenst,mef,llp@cs.princeton.edu>

1 Introduction

During the last year, the PlanetLab implementation has shifted from a distributed design, where both the node and PLC maintained authoritative state about slices, to a more centralized one, where the management and slice naming and creation authority are integrated, and the node serves only as a cache for state maintained at PLC. As well, OneLab Europe[1] recently agreed to federate their infrastructure with PlanetLab; they will act as an independent management authority (MA), and share a cooperating slice authority (SA) with PlanetLab. Finally, Emulab continues to develop their portal to PlanetLab, using new mechanisms PlanetLab provides for delegation. A user with an account at both PlanetLab and Emulab can delegate Emulab as the principal to instantiate their account on specific nodes. Emulab serves as an alternative slice creation service for their users, an operation typically performed by PLC.

As PlanetLab continues to grow in geographic scope, we expect other autonomous regions, like OneLab, to act as independent MAs. Of course, many other testbeds already exist, such as VINI[2], Emulab[5], and Everlab. These testbeds often provide access to novel resources not available on other testbeds. As these testbeds become more common and users more aware of them, users will expect either (a) their favorite testbed to integrate the popular features of the others, or (b) the ability to seamlessly access the resources of any testbed. The first case is not tenable in all situations, and in the second case, a single SA could serve users by peering with the MA of the various projects.

PlanetLab developed MyPLC in order to enable others to establish their own testbeds with which we could federate[4]. But, MyPLC can at best only serve as a reference. Much the same way as ARPANET gave way to the Internet, diverging interests seem inevitable[3]. For instance, the OneLab project has already forked parts of the PlanetLab code to further their goals (e.g., support for wireless networks).

Since the ultimate goal for PlanetLab is to enable cooperation among all principals by finding a general architecture, common API, and shared abstraction for testbed federation, we propose a generalized architecture based on our experiences over the last year. The following traces the history of several recent implementation deci-

sions, showing how they can be expressed as interactions between management and slice authorities, and finally suggests a general architecture derived from the common aspects of these specific problems.

2 Models in Practice and Design

PlanetLab has anticipated the federation of infrastructures, yet currently our implementation is largely centralized. Today we are engaged in the administrative and development steps needed to federate with OneLab and to provide better support for user delegation with Emulab. The following outlines how we are approaching a generalized model for federation from the starting point of a centralized implementation.

PLC: The private PlanetLab package, MyPLC, bundles two logically independent functions, the Management Authority and Slice Authority. Earlier work has defined the trust relationships between these components[4], and the following speaks more from an implementation perspective.

Figure 1 represents the PLC database separated by the logical MA and SA components. As indicated by the direction of the arrows, Users express their slice preferences to the SA and the Nodes receive and act on announcements from the MA. The MA maintains the authoritative list of nodes, what sites they are associated with and relevant contact information for maintenance. The SA also maintains a possibly disjoint list of sites and users who can create or use slices. The SA is the authoritative source of slices in the system. The MA advertises the node list to the SA, for assignment to slices. The values for which the SA and MA are not the authoritative source is strictly a cache.

Management Delegation: In the old PlanetLab implementation when the authoritative state maintained by PlanetLab was distributed across both PLC and the nodes, Node owner management (NOM) kept ultimate control over the machine's resource allocation with the node owner and recorded these preferences at the node. Figure 2b illustrates the parallel and conflicting paths from the MA and NOM to the node. Because the node acted as an authoritative repository for resource allocation state, it was possible for a mismatch between the owner's preferences, the user's expectations, and the

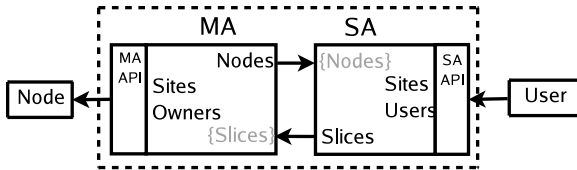


Figure 1: The PLC database divided by MA and SA.

record maintained by PLC. The solution we adopted was to centrally locate all owner policy into the MA. Looking forward, though, it may be appropriate to place the policy close to the node, outside of PLC, but clearly the path from root MA to the node should be unique.

Slice Delegation & Independent Slice Authority: Emulab requires synchronous slice creation from the time a user submits an experiment to the time it runs. By default, PlanetLab offers an asynchronous mechanism for slice instantiation. To address this conflict, Emulab has implemented an alternative slice creation service (SCS) that operates on behalf of the user, a service typically provided by PlanetLab. In doing so, the slice (at creation time) *delegates* its instantiation to Emulab. Figure 2c illustrates the approach taken by Emulab.

Currently, Emulab is limited by the total number of slices they can create (10), rather than the number of slivers actually instantiated across all machines. To work around this allocation policy, Emulab has requested the ability to *create* slice names for other sites within the PLC SA. However, this violates the hierarchical slice authority: only a site may create its slice names. After the name is created, the instantiation, provisioning, environment configuration, and access may all be delegated. So, for now, the user must interact with both entities, the PLC SA and the Emulab SCS.

If Emulab were a child SA to PLC’s SA, then by virtue of being an SA, they could create arbitrary slice names within their namespace. Alternately, they could become a second, independent SA as illustrated by figure 2e, which would include a unique slice or sliver limit, independent of a parent SA.

Independent Management: We are currently in the process of federating with OneLab Europe. Part of the agreement entails that OneLab will take over responsibility for the machines falling within their geographic region. Because they are basing their testbed on MyPLC, they also have a SA that is independent of ours. Because the SA namespace is currently flat, there is no distinction between PlanetLab’s SA and OneLab’s SA. This has led to temporary conflicts in associating a user with the SA a user belongs to. It should be possible for a user to join both SAs. While this will be worked out in time, an alternative solution would be to run a single SA that interacts with both MAs, allowing users who join the single SA to access machines operated by both MAs. This approach is exemplified by figure 2d.

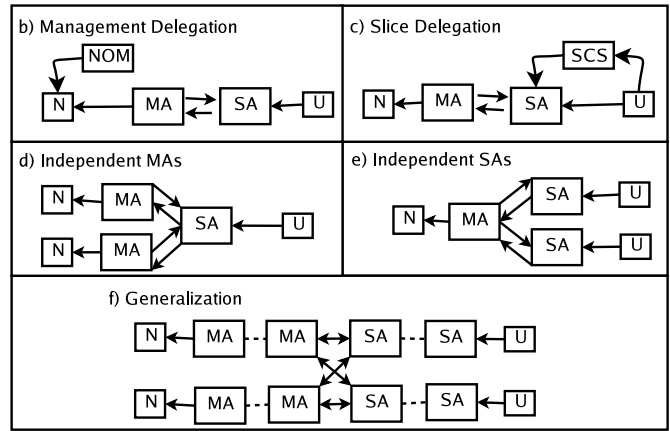


Figure 2: Five Models for Federation Based on the PLC MA/SA split.

3 Generalization

Finally, figure 2f attempts to unify the components of the previous four. SAs may be nested to an arbitrary depth where each parent acts as a delegate for the child. And, on the MA side, owners may act as their own MA, that is a child of a MA higher in the tree toward the root MA.

Additional discussion will demonstrate that the critical trust relationships codified within PLC are maintained by the split between MA and SA, that the scalability of the architecture remains good when the number of peers is large, or that the data shared between different authorities remains isolated and bounded, such that the actions of one peer will not compromise the state of another. Once these issues are addressed we will have a solid architecture over which more complex policies are applied for resource allocation, the expression, exchange and enforcement of acceptable use policies, as well as where the components for enforcing policy are placed within this architecture.

References

- [1] Onelab project, 2007. <http://www.one-lab.org>.
- [2] Andy Bavier, Nick Feamster, Mark Huang, Larry Peterson, and Jennifer Rexford. In VINI veritas: Realistic and controlled network experimentation. In *Proceedings of ACM SIGCOMM 2006*, Pisa, Italy, September 2006.
- [3] David D. Clark, John Wroclawski, Karen R. Sollins, and Robert Braden. Tussle in cyberspace: defining tomorrow’s internet. In *SIGCOMM 2002*, pages 347–356, New York, NY, USA, 2002. ACM Press.
- [4] Larry Peterson, Andy Bavier, Marc E. Fiuczynski, and Steve Muir. Experiences building planetlab. In *Proceedings of the 7th USENIX Symposium on Operating System Design and Implementation (OSDI ’06)*, Seattle, WA, November 2006.
- [5] Brian White, Jay Lepreau, Leigh Stoller, Robert Ricci, Shashi Guruprasad, Mac Newbold, Mike Hibler, Chad Barb, and Abhijeet Joglekar. An Integrated Experimental Environment for Distributed Systems and Networks. In *Proc. 5th OSDI*, pages 255–270, Boston, MA, Dec 2002.