Building Toward an Open and Extensible Autonomous Computing Platform Utilizing Existing Technologies

Third IEEE WoWMoM Workshop on
Autonomic and Opportunistic Communications (AOC 2009)
June 15, 2009 - Island of Kos (Greece)
http://cnd.iit.cnr.it/aoc2009/

Phil Cryer
www.philcryer.com

Missouri Botanical Garden - Saint Louis, Missouri, USA
www.mobot.org

Abstract

Today's continuously growing amount of data, paired with the expanding hardware requirements such stores demand, stresses currently overwhelmed information technology staff, as their responsibilities multiply with every new server. There is a real need for autonomous computing systems that can look after themselves and demand far less human intervention than previously required. While complete autonomous computing systems are being theorized about, much of its promise can be realized today by using multiple, existing open source software applications, a basis that allows for the ultimate in flexibility and customization. Building a distributed system using autonomous computing allows for an administration protocol which provides true opportunistic communication between autonomous nodes, as any of them could serve as authoritative, or querying members, depending on their current knowledge of required tasks. All of which allows us to provide an open and extensible autonomous computing platform utilizing existing technologies.

Network Topology

When building an open, autonomous computing environment, an overall architecture plan is the first step to decide upon. Since a star, ring or tree network topology all leave you with a single point of failure, utilizing a distributed architecture network should be considered at this time, since it solves many issues, and best supports the autonomous computing ideal. Even if there are only two system nodes, it is highly preferable to have them both taking care of each other rather than administrating both manually. Identically created system nodes can pass along commands to allow its autonomy to be passed along to others in the group. Nodes are able to then able to contribute at various levels depending on which tier of authorization they are assigned to, and can adapt and change as instructions cascade through the group. Being cognizant of the fact that both budget and information technology groups are being strained as requirements for systems increase, a distributed administration environment is the best way to combat both factors.

Operating System Software

While many different Linux distributions could be used, I highly recommend standardizing on the Debian GNU/Linux[1] distribution. It is one of the oldest Linux distributions, having been launched in August 1993, and is known and respected for focusing on stability and security, while being known for strict adherence to free software philosophies. It also offers the Debian package management utility, APT[2], which can be automated to take care of basic tasks in terms of checking for security updates, and automatically resolving dependencies when applying such updates. The abilities of APT are key in providing a secure autonomous system, as any security updates can be applied automatically, with notification emailed to an administrator. Additionally, Debian is the basis of one of the most popular Linux distribution today, Ubuntu Linux[3], which is quickly becoming one of the most used Linux distribution, a commentary to its quality and flexibility.

Implementation and Installation

When considering software it is critically important to understand how the software will be implemented so that automating the install, and eventual updates, is possible. This step allows a disaster recovery plan that causes the shortest amount of downtime, and reduces the need for support staff to manually re-install the system. The Debian distribution has a specialized tool to take care of this exact task, which is called preseed[4]. Using preseed it is possible to have an installation procedure that automatically downloads and installs of the operating system, after being booted off a standard USB thumbdrive. Using this not only insures that only what is needed is installed, but also that only the latest, and most secure, versions of software are applied. Examples of this have reduced base installation time 20 minutes or less[5]. Post install scripts can extend this functionality, allowing the system to automatically set up networking, needed services and limits to conform with standards chosen by the administrator. For example, it's beneficial to have a new system ping a central server so that it can understand the network topography and global location, using global locating software GeoIP[6] and Internet Traffic Report[7] for future monitoring and global network trending. Having a new system perform this type of 'lint test' would provide it with crucial information that it can use in the future.

Shared System Administration

During setup it is simple to install and instruct scripts not only to take care of the local system, but of remote systems as well. Two main functions of this administration are keeping the system configured and up to date with security releases, while also ensuring the system and services are available.

Shared System Administration - Configuration

The distributed configuration application Puppet[8] helps to accomplish the goal of a hands-off, automated infrastructure. Created by a former developer of Cfengine[9], which is an earlier configuration application, its features are an improvement over previous systems, with simplicity and flexibility over a variety of platforms its focus. With Puppet a network with two nodes, or hundreds of nodes, would act exactly the same, it keeps all systems in sync. Utilizing scripts, that it refers to as recipes, Puppet is able to handle changes to configurations, packages, applications, whenever they are needed. While Debian itself will take care of log rotation and other core system duties, another crucial aspect of keeping Debian systems up to date with security releases is by using a wrapper for APT called apticron[10]. This works via cron, which Debian runs automatically at predefined intervals, and when run will update the system's current cache of files and available updates. By setting the configuration file sources.list to only monitor security updates, apticron can notify an administrator by email or SMS when there is a security update available. Since Debian's security history has been excellent, I have no problem setting apticron to automatically install any security updates as soon as it finds them, thus allowing for a hands off, autonomously updating system that alerts me after it performs such an action. With this method in place, any security issues on a Debian system will likely be updated by the time most hear of the vulnerability.

Shared System Administration - Monitoring

System monitoring is provided by monit[11], which is a utility for managing and monitoring, processes, files, directories and file-systems on a UNIX system, designed as an autonomous system that does not depend on plugins nor any special libraries to run. Once configured monit can monitor and manage distributed computer systems, conduct automatic maintenance and repair and execute meaningful causal actions in error situations. Basically monit will watch processes, check system resources and react accordingly when something amiss is found, and alert an administrator that it has taken action. When having it watch permissions it even allows it to perform basic intrusion detection duties, and its abilities are not confined to local systems, it can just as easily monitor remote systems, allowing for an external view of other server's processes. Having used monit for years, I can attest to its abilities, and often when using monit in concert with the above mentioned apticron I can go months without having to touch a production system since I know if there is any issues thanks to these two, simple applications.

Distributed Data

There are a number of distributed filesystems in use, and by utilizing them a system becomes more autonomous since its data may be spread across multiple systems, thus making its existence far less important that if it held all of the data. Examples of these systems range from a few nodes, to thousands of nodes in a cluster. One of the highest profile systems is called Hadoop[12], which is a distributed computing system used by Yahoo, which includes a distributed filesystem called Hadoop Distributed File System (HDFS)[13]. This is referred to as an open source clone of Google File System (GFS)[14], which is Google's similar, proprietary effort. HDFS is designed to scale to petabytes of storage, and run on top of the fileystems of the underlying operating systems. This lack of any exotic requirements further puts it in the autonomous camp; it can be run across a variety of platforms, and again, with data spread across so many nodes, the original node is not important for the integrity of the data to remain constant. Other examples of distributed files systems that are ready for large scale use are GlusterFS[15] and Lustre[16], which was recently acquired by Sun Microsystems[17].

Hardware

A final key consideration when building an autonomous system is to consider the hardware, and the impact of the server when it is taken out of the group of servers. For this application, standard x86 or x86_64 bit servers with standard PCI, AGP and SATA interfaces become quite attractive. Not only are the prices for such systems far less than proprietary UNIX servers, another big benefit about staying with generic, off the shelf x86 PC hardware is that any information technology department, even a very small one, will be comfortable swapping out components if they ever need to, with basic parts you can buy anywhere, or order online. These are all common pieces of hardware, easy to replace in every way.

Conclusion

While there are plenty of theories that one day might make complete autonomous computing a reality, it's important to understand what can already be accomplished today. Using multiple, existing open source software applications, allows for the ultimate in flexibility, and is ideal to design and build a system that many autonomous configurations. All of which provides an open and extensible autonomous computing platform utilizing existing technologies.

References

[1] Debian GNU/Linux, *An operating system (OS) for your computer using the Linux kernel,* http://www.debian.org/

[2] Apt HOWTO, *This document intends to provide the user with a good understanding of the workings of the Debian package management utility, APT,* http://www.debian.org/doc/manuals/apt-howto/

[3] Ubuntu Linux, *Ubuntu is a community developed, Linux-based operating system that is perfect for laptops, desktops and servers,* http://www.ubuntu.com/

[4] Preseed, *Contents of an example preconfiguration file for Debian,* http://www.debian.org/releases/lenny/example-preseed.txt

[5] Aaron Toponce, A*utomating Debian/Ubuntu Installs With Preseed,* http://pthree.org/2008/05/20/automating-debianubuntu-installs-with-preseedanother

[6] GeoIP, *GeoIP provides businesses with a non-invasive way to determine geographical and other information about their Internet visitors in real-time,* http://www.maxmind.com/app/ip-location

[7] Internet Traffic Report, *The Internet Traffic Report monitors the flow of data around the world,* http://www.internettrafficreport.com/main.htm

[8] Puppet, *The Puppet framework provides a means to describe IT infrastructure as policy, execute that policy to build services then audit and enforce ongoing changes to the policy,* http://reductivelabs.com/products/puppet

[9] Cfengine, *Cfengine is a policy-based configuration management system,* http://www.cfengine.org

[10] Apticron, *Automatic package update nagging with apticron,* www.debian-administration.org/articles/491

[11] Monit, *Monit is a utility for managing and monitoring, processes, files, directories and file-systems on a UNIX system,* http://mmonit.com/monit/

[12] Hadoop, *Hadoop is a free Java software framework that supports data intensive distributed applications,* http://hadoop.apache.org/core

[13] Hadoop Distributed File System (HDFS), *Hadoop Distributed File System (HDFS), is an open source Java product similar to GFS. It is designed to scale to petabytes of storage, and run on top of the fileystems of the underlying operating systems,* http://hadoop.apache.org/core

[14] Google File System (GFS), *Google File System (GFS) is a proprietary distributed file system developed by Google Inc. for its own use. It is designed to provide efficient, reliable access to data using large clusters of commodity hardware,* http://en.wikipedia.org/wiki/Google_File_System

[15] GlusterFS, *GlusterFS can scale to multiple Peta bytes and 100s of GB/s throughput, can sustain 1 GB/s per storage brick over Infiniband RDMA and can self-heal itself on the fly,* http://www.gluster.org/

[16] Lustre, *Lustre is a scalable, secure, robust, highly-available cluster file system. It is designed, developed and maintained by Sun Microsystems, Inc.,* http://lustre.org

[17] Sun Microsystems, *Software company producing Java and Solaris Operating System for Sun Hardware,* http://sun.com