Real-World Multipurpose Network Infrastructure

Brian Parks
bparks@vast.uccs.edu

Anthony Magee
macawm@vast.uccs.edu

Abstract

We present a description of a live working network infrastructure designed to support many clients and services in a dynamic work environment. We cover some of the issues and design choices that must be solved in a short time frame on a limited budget. The authors' combined experience in different environments provides us the opportunity to share a wide variety of experiences and problem solving paths from multiple perspectives. In particular, we discuss some of the requirements of a flexible network environment, as well as some of the security and policy considerations necessary for its smooth, uncompromised operation.

1. Introduction

It is difficult to imagine a time without the Internet. In fact, very often the consumers of the Internet and the World Wide Web take it for granted. However, networks do not run themselves and the design of networks plays very strongly into its effectiveness in the situation for which it is used. On top of the technical concerns of simply running a network, the end users and potentially political motivations need to be balanced.

To that end, we discuss the nature of some of these concerns, drawing on our experience as System Administrators of networks designed to serve various purposes at various places and modeling our current discussion after the infrastructure we maintain at the Vision and Security Technology (VAST) Lab[1] at the University of Colorado at Colorado Springs. We discuss our approaches to solving these issues in the context of a multipurpose network infrastructure, paying particular attention to the requirements and pressures that both management and the end users will place on the network managers as *their* requirements and needs change.

The remainder of this document is laid out as follows: Section 2 outlines various purposes for a network infrastructure, in particular the purposes for which the VAST Lab network is tailored, with a brief discussion of other situations with which the authors have experience from previous incarnations of the network and from prior experience. Section 3 outlines some of the services employed to fulfill the purposes introduced in Section 2. Section 4 discusses the types of pressure involved and how a network can be configured to maximize versatility and we conclude in Section 5.

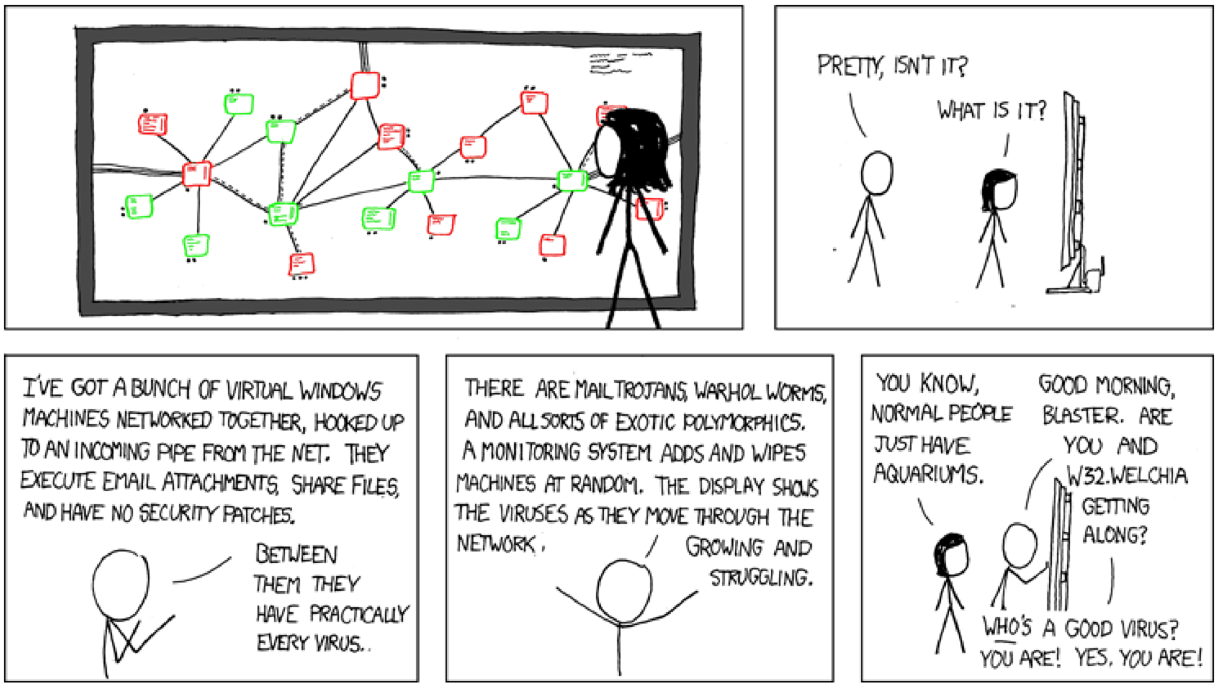
[1]http://vast.uccs.edu

2. Purposes

Our network at the VAST lab is a very fluid environment and as such requires substantial interaction on a daily basis to keep everyone happy. The reason for this fluidity is that we have on average eight *power* users that have high demands for interoperability, flexibility, and reliability. These users along with about fifteen others that do not demand special services are constantly wanting something else or something new.

Some of our network's purposes are:

- Storage - This is our primary reason for running a dedicated cluster of servers. We are research oriented and can generate many gigabytes of data on any given day. We also provide, as a courtesy, a large amount of storage for users' general storage. This storage comes in many forms including networked hard disk, tape, and portable disks.
- Processing - Generating data is our secondary responsibility because more often that not a user's single machine cannot run CPU intensive jobs at a quick enough pace to meet deadlines. This means that we are constantly adding and upgrading our high density CPU machines to be as fast as possible. With this many users however, scheduling who can use CPU time and at what time can become challenging.
- Connectivity - There are two types of connectivity that we must support, namely open lab access and restricted external access. This implies that a user working at a lab terminal or personal computer should have no encumbrances for access and


PRETTY, ISN'T IT?
WHAT IS IT?
I'VE GOT A BUNCH OF VIRTUAL WINDOWS MACHINES NETWORKED TOGETHER, HOOKED UP TO AN INCOMING PIPE FROM THE NET. THEY EXECUTE EMAIL ATTACHMENTS, SHARE FILES, AND HAVE NO SECURITY PATCHES.
BETWEEN THEM THEY HAVE PRACTICALLY EVERY VIRUS.
THERE ARE MAILTROJANS, WARHOL WORMS, AND ALL SORTS OF EXOTIC POLYMORPHICS. A MONITORING SYSTEM ADDS AND WIPES MACHINES AT RANDOM. THE DISPLAY SHOWS THE VIRUSES AS THEY MOVE THROUGH THE NETWORK,
GROWING AND STRUGGLING.
YOU KNOW, NORMAL PEOPLE JUST HAVE AQUARIUMS.
GOOD MORNING, BLASTER. ARE YOU AND W32.WELCHIA GETTING ALONG?
WHO'S A GOOD VIRUS? YOU ARE! YES, YOU ARE!

that a reasonable amount of resistance be encountered by hackers trying to compromise the system while still allowing relatively easy access for valid users. All of this is to provide users access to the two above resources.

- Reliability - Our system is not one that needs a supremely high uptime, but it is nice to provide our users with consistent accessibility schedule. Given that we change so many aspects of our system frequently this means that we get feedback on a regular basis. Most of this feedback is regarding some feature that used to work and suddenly stopped working. One of the goals to reduce this feedback is to increase our reliability; a task that we are always working on and thinking about.

- Safety and Security - Providing safety to our system is crucial to its reliability; especially from our own users which seem to be able break things at will. Protecting the physical and virtual network from users is a challenging prospect at best. Users are always trying to do something they have no need to do or they are simply asking us to do things that will break other aspects of the network that they have not considered. This in turn affects the policies that we have put in place to keep users' data safe from accidental deletion or modification and to keep any private data we may have secure from malicious attack.

3. Services

Offering a basic set of services for our users is the very least we can do as administrators. Yet in our research environment basic services are rarely sufficient and must be supplemented with one-off solutions to problems that arise on a daily basis.

The largest challenge outside of actually getting solutions in place is making sure that the information is captured. This means that in our team of administrators one person's actions must be recorded in some fashion for two important reasons. First, if another team member wants to verify or modify any recent network or system change it must have been recorded or else it will be quickly forgotten. As the senior members of the team we are often going back over our juniors work to verify that no security holes have inadvertently been opened by their actions. Second, it is vital to the networks long term health that precise records are in place to show the problems encountered and their solutions. Without this we would be constantly "reinventing the wheel" which is not acceptable given out limited budget.

Our budget is what you might call flakey, or more appropriately, nonexistent. Due to the nature of our "work" getting a network and systems budget is difficult and one might say that these things should just be there ready for use. But, in reality we must do everything ourselves. This means building machines, installing operating systems, running wires, asset tracking (virtual and physical), organization, plugging security holes, and a

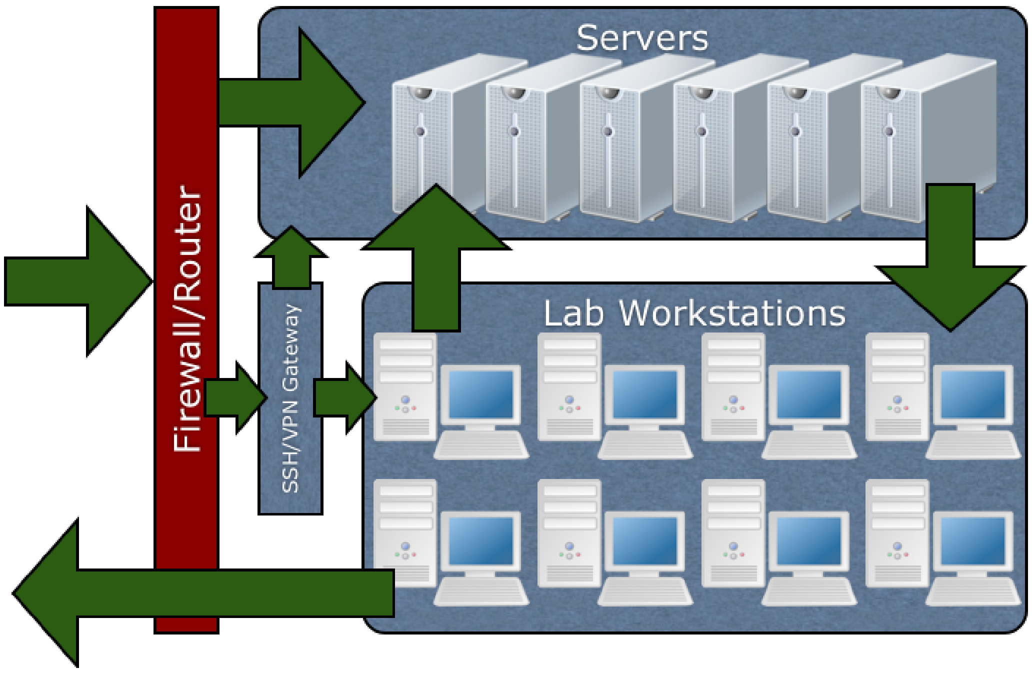


Figure 1. High-level diagram of the VAST Lab network

plethora of other ancillary jobs with their own special hats. And, what this means is that everything must be done as cheap and reliably as we can manage.

This in turn effects which and how many services we can offer and accommodate. Currently we have over thirty regularly used services that help our users generate, manage, and have reliable data stores, as well many others that they have never seen but a crucial to the whole system's well-being. We will cover some of these services below.

3.1. Network

By "network," we mean the essential services required to keep the network running. This may sound like a recursive depndency, but in practice it is not. The services that fall into this category are services such as DHCP, DNS, and directory services. For the first two services, we use the *de facto* standards `dhcpd` and `bind`, respectively. However, many implementations exist for directory services, each promoted by its share of the System Administration community, out of which we have chosen the Lightweight Directory Access Protocol (LDAP).

Although we employ DHCP (Dynamic Host Configuration Protocol) for our IP address and hostname assignment needs, most of the hosts in our lab have static routes in the `dhcpd` configuration. This makes it far easier to monitor and analyze traffic in the event that something fishy appears. In addition, this allows for easy assignment of meaningful DNS hostnames to most machines in the lab.

DNS plays a fairly significant role in our network infrastructure. We try to give meaningful names to most machines to divide them into classes based on naming scheme. For example, most of our servers have names of servants (a common naming convention among System Administrators). These names are then aliased (CNAME in DNS-speak) to functional names, such as ldap.vast.uccs.edu and dns.vast.uccs.edu.

Directory services provide a consistent set of authentication tokens across a multitude of machines, and we employ this on most of our servers (unless there is a good reason not to) and on several of our lab workstations. This allows anyone with a lab account to log on to any machine that uses this method as themselves and have the appropriate rights and privileges. Through a set of additional features, we can also manage local machine privileges from a central location.

3.2. Storage

As a research lab, the primary requirement is storage. As mentioned above, we perform experiments on large datasets and end up generating large datasets as results. Unfortunately, given a limited budget, this means storage options are small and fragmented. Each machine has its paucity of storage in various RAID configurations (to be described below) which must be appropriately allocated for various tasks.

In addition, each user requires his or her own space to use, which should contain itself to a reasonable limit. We enforce this through the use of NFS soft quotas, which warn the user for up to a week or so after they exceed it. After that, they can only delete files from the directories they own before their share of the filesystems is again usable.

Most of our machines contain several disks in a RAID5 configuration through a card or on-board hardware RAID controller. This provides for easy configuration of the operating system on top of it (it is oblivious and sees the array as one large disk), while still allowing for painless redundant storage. However, this is still subject to hardware failure, as we found out the hard way not too long ago when two disks died in one of our servers.

In order to prepare for such hardware failures, it is necessary to keep spare disks on hand to replace disks as they die. It is guaranteed to happen; it's just a matter of when. In addition, it is always useful to have a spare RAID controller lying around for each machine that uses hardware RAID in case the one in use fails.

In addition, one can not assume that simple redundancy will be enough. As in the case of two disks in a RAID5 dying, there will be extraneous circumstances that require additional backups. As old as tapes are, they are still the best option. We have a saizable tape changer at the VAST lab and everywhere the authors have worked as System Administrators has also had a tape changer, usually coupled with the UNIX program `amanda`, regardless of host operating system. Of course, tape drives being hardware, there is always a chance that they, too, will fail. However, by this point, there are several levels of redundancy and the probability of all failing at the same time is extremely low.

3.3. Web Services

While we are, first and foremost, a research lab in the field of computer vision and biometric-related security, we do require a certain amount of web presence. Our advisor, Dr. Terrance Boult, is also heavily involved in several other organizations on campus including the El Pomar Institute for Innovation and Commercialization[2] and the Bachelor of Innovation[3] Program, so our "Web Team" is tasked with the upkeep of the relevant web sites as well. Thus, we have a fairly extensive web infrastructure.

Our setup essentially includes four classes of web servers. The first class is the production web server. This is a single machine that gets touched only periodically to push new code from the testing phase to the live phase. Our other semi-live server runs all of the lab's internal web sites and is, for the most part, not part of any formal artifact management sequence (that is, we develop and test on the same machine).

The third class of machine comprises a single machine used to test developed code that will (hopefully) eventually end up on the live server. This machine has a configuration and architecture **identical** to that of the live server so that live code breaks as infrequently as possible.

Finally, we have a series of developent machines. These are actually Virtual Machines that have as similar a configuration to that of the testing server as possible. Each web developer has his or her own virtual machine to do development on to facilitate the distributed version control paradigm and avoid stomping on each other when each developer is working on a different feature-set with a potentially different timeline. This has been the biggest headache in the past few weeks, as it is certainly a big change from doing development directly on the test server (the method for web development previously employed).

This infrastructure is modeled after systems used in industry for environments supporting a large number of moderately-sized web applications. Each machine in all classes has its own LAMP (Linux, Apache, MySQL, PHP) stack to isolate the phases of development while still maintaining continuity between phases (i.e.: it's much more cost-effective to spend time fixing bugs than reconfiguring software after a move from one machine to the next).

3.4. Remote Access

While our network comprises an entire class C subnet (128.198.147.0/24), we don't necessarily want all of those machines accessible from the outside world on every possible port. In fact, since most of the machines in the lab are personal workstations, we don't want them accessible from the outside world *at all*. Some of the servers run read-only services (web, for instance) which are safe to make world-accessible. However, the more volatile services (SSH, for instance) have further restrictions handled by our router and firewall machine.

In a situation such as this where the majority of machines do not need to be accessible from the outside world, it is far easier to block all incoming traffic and then make exceptions to this rule. Outgoing traffic (by

[2]http://epiic.uccs.edu/
[3]http://innovation.uccs.edu

this we mean traffic over a connection initiated by a machine inside our network) is enabled for all machines, save one (that machine also has the additional security measure of having no default gateway set, so that the only traffic it is capable of delivering is traffic internal to the network). Exceptions to the ban on incoming traffic is done by service, so machines in the "web server" group are permitted to receive traffic on ports 80 and 443. Machines in the "mail server" group can receive traffic on ports 25 and 587 and machines in the "DNS server" group can receive traffic on port 53. Various other services are configured and allowed as needed.

Note how SSH (port 22) appears to be strictly disallowed from all machines in our network. This is not quite the case, but we like the appearance of blocking all SSH traffic. We do have one machine to which SSH is allowed; it is just an SSH gateway and does not provide direct access to any of the services we provide so that, in the event it does get hacked, the damage an intruder can do is minimal. This is one of two ways that legitimate network users can access our resources from the outside world. The other is through an OpenVPN-based Virtual Private Network.

We provide the VPN service to our users in the event that SSH is insufficient. We use a key-based system to maximize security, though we recommend against the use of the VPN, since the within-network endpoint is running on an old 1U Intel rackmount server from several years ago. In addition, the VPN induces a large amount of overhead, so we recommend using SSH whenever possible.

3.5. Computation

In order to provide for computation, a machine generally needs adequate RAM and processing power. This situation generally is very easy to configure, as the end users are running their own programs. Thus, it needs to be a relatively controlled environment so that multiple users don't end up stomping on each other. If user home directories are mounted from elsewhere (such as by using NFS) and the only files present on the system are those required to run the system and computation-intensive experiments, this is usually sufficient. In the event that the end users are untrustable or you don't want to take any chances, using SSH chroots is always an option. Of course, since computation is the main goal of these machines, also running any other computation-intensive services on a compute server is suboptimal.

In the VAST Lab, we don't have nearly enough machines to separate every service or set of services onto their own machines. However, most network services aren't very computation-intensive and thus can happily coexist on the same machine as computation-heavy user programs.

4. Flexibility/Versatility

Requirements are always changing. This is a classic theme in Software Engineering that has given rise to the so-called "agile" methods and it is no different when it comes to Network Management. One could view the whole of network management as one big software engineering project that tends to be lopsided toward responding to changing requirements more than it involves developing. Regardless of perspective, the fact remains that the network designed today will not necessarily be the same network required to maintain productivity several months from now. Thus, some degree of flexibility must be maintained.

However, a network should not strive for the goal of being as flexible as possible so much that focus on maintaining a functional network is lost. Thus, one must take into account the direction that the end users and management are pushing and anticipate, to a certain degree, the design decisions that will need to be made in the near future. A balance must be struck between designing parts of the network for each new requirement in isolation from the rest of the network and the opposite extreme of making it possible to fulfill the radical requirements at the expense of making the small changes difficult to implement.

As a research lab, the VAST Lab has its fair share of changing requirements. Contract to contract (even task to task) the requirements change as to what computing and storage resources will be required. Given the length of time that the lab has been around, we've learned how best to do things and how explicitly *not* to do others. We have found that, in general, the best strategy is to design things to be as modular as possible and reduce the number of interdependencies between services and machines. This is a difficult ideal to acheive, especially when ease of configuration and a high expectation of security are thrown into the mix.

5. Discussion

Since the network described above is, first and foremost, a functional network, with all the necessary bounds of a real-world situation such as a limited budget, limited space requirements, fire codes, and other such facts of life, its implementation differs in several ways from what could be considered the "ideal." The decisions made to omit these features is a question of economy: how much network is worth the trouble on

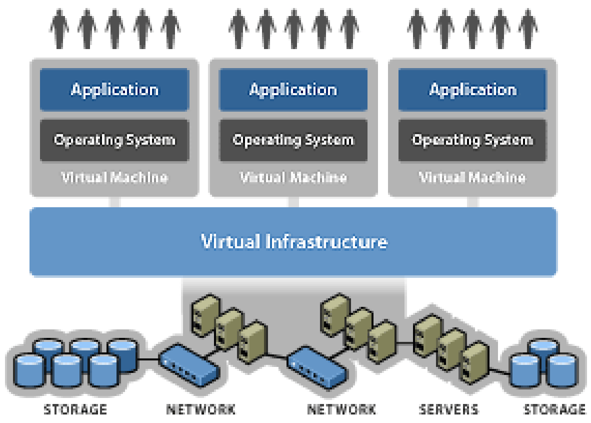


Figure 2. VMWare Virtualization Infrastructure

such a limited budget?

One such omission is physical redundancy. While redundancy is certainly better than downtime in the event of hardware failure, it requires more equipment (in general. See below, "Virtualization") and thus a larger budget. It essentially boils down to an evaluation of the extent to which the services provided are critical versus the cost to keep them running 99.99% of the time. Thus, it may be more cost-effective to keep a few extra RAID cards on hand for the major file servers than to have them go down for a few weeks (reasonable), but it may not be as cost-effective to have an additional router configured as a failover when the majority of network outages are due to circumstances beyond your control that would affect *both* routers.

Another area where we stray from what some would consider "ideal" is the age of our equipment. Let's face it: older hardware is potentially more likely to break. However, at a certain point, hardware that has been around for a while is tried and tested. Thus, older hardware is a cost-effective option to maintain a moderately large network on a low budget. Even more so, if the networks to which you connect have a significantly lower capacity, you can get away with running old hardware and still not being the bottleneck.

5.1. Virtualization

Separation of services is generally considered to be a Good Thing in the System Administration communities in terms of security, configurability, and the avoidance offailure due to software issues. However, this is not always possible with limited hardware and limited budget. Virtualization has taken the datacenter by storm in recent years to solve this problem: one powerful server can easily virtualize several less-powerful servers, each providing one or two services to the network. Similar goals have been achieved through the concept of zones (Solaris, by Sun Microsystems) and domUs (Xen), which virtualize part of the operating system at a kernel-level, while still maintaining the usability of the operating system running on the bare hardware.

The authors do not recommend virtualization for large-scale service deployment for several reasons, outlined below. This comes with the caveat that virtualization does have a useful time and place, such as transient services that do not require high uptime or responsiveness.

1. Virtualized services go through at least two levels of context switching, depending on the hypervisor. The first level of context switching is inside the guest Operating System running the desired service. The service itself must share resources with the operating system, and this operating system must share resources (as allocated by the hypervisor) with other virtualized operating systems on the host machine.

2. Virtualized systems have a limited ability to make use of the base hardware. This may not be desirable, for instance, in a situation where a storage server needs to make use of a hardware RAID card independent of the other guest operating systems, or if a specific aspect of the CPU is beneficial or necessary for computation.

3. Virtualization technologies tend to be oriented around a particular architecture, specifically the x86 architecture. This presents security issues, as exploits are particularly common for x86, especially for machiens running web-related services. This means that the best way to avoid suffering from x86-related exploits is by not running virtualized hardware.

This is not to say that virtualization is necessarily *bad*, but it is certainly not the cure to the all-too-common problem of machine and funding shortages. Services that can be started when needed and stopped when the end user is done with them are an excellent candidate for virtualization. In addition, services that need to be isolated and don't have a high responsiveness requirement are also candidates for virtualization.