Personal Home Cloud Systems: An Exploration of Options

Jasper Wu
Spring 2018

Modern society is trending towards interactions through social media and, as a result, is placing personal information publicly on the Internet. For convenience, many people share information through social media giants such as Google, Facebook, Instagram, Snapchat, WeChat, Line, Flickr, and many others. Yet such services do not provide flexibility and, more importantly, ownership over one's own data.

We propose a more flexible system that not only encompasses the same use cases as enterprise clouds but also ensures privacy; we call such a system a personal home cloud.

In this paper, we propose a design for a personal home cloud system using common-off-the-shelf software and hardware. We begin by presenting the characteristics we use to evaluate potential components for the system, and then we present the results of the evaluation. Finally, we present the system we designed, its advantages, and its disadvantages.

Design

Our goal when designing this system is to mimic enterprise cloud solutions. Using Google Drive as our example, consider a user completing the task of signing in and navigating to a file. Though this task may seem simple, the user expects it to go smoothly, while various tasks are handled by machines behind the scenes. First, the user expects Google.com to access the Google they expected and not direct to another site. Then there is authentication and authorization of the user. Once logged in, the entire interaction is expected to be responsive and generally perform smoothly. Behind the scenes, employees at Google are maintaining the entire system and making sure it is never down. Finally, the user expects Google, as a trusted company, to safely and securely store their data.

Considering this example, we recognize that any cloud system requires these components: a static endpoint, authentication and authorization, encrypted storage, and administrative control. At first each component may seem simple and straightforward, but they are often complicated in the context of a personal system.

Characteristics

Formulating proper characteristics is essential to the evaluation of any system. In evaluating components for our feature-complete personal home cloud, we recognize that it may not be possible for all of our characteristics to be completely met at once. For that reason, we allow characteristics to be evaluated on a scale, so that some solutions where characteristics are "almost met" remain in consideration.

Security

Security is important in any system connected to the Internet, since such systems are vulnerable to outside attacks. For our personal home cloud, we are most concerned with protecting user data through authentication, authorization, and encryption. For our purposes, we need any security measure we implement to be both easy to deploy and easy to use. Note that while it is always important to to provide the necessary security whether in an enterprise system or a personal cloud, we also realize that a personal system has a narrowed scope of likely attacks. For instance, attacks such as DDoS and phishing would be unlikely.

Performance

Performance is the speed at which a service can be provided. For our personal home cloud, we care about performance because it factors directly to the user experience. We need to evaluate this characteristic in two pieces: internal and external performance. In our case, performance is determined by the quality of the home LAN and the Internet Service Provider (ISP). Max internal performance in a LAN could reach Ethernet gigabit speeds or even multiple gigabits if multiple Ethernet connections were available. Extending a personal cloud out to Internet on the other hand is limited to the ISP service. Consumer Internet speeds vary and are not completely reliable, but overall performance has been increasing every year.

Reliability

In our system, we are concerned with reliably storing user data and the availability of the system. When using a properly designed service, this is seamlessly handled behind the scenes and not a concern. In personal clouds, easy solutions do exist to keep data intact, such as Redundant Array of Independent Disks (RAID). Using RAID 1, for instance, to mirror data across two storage drives guarantees that the data will be safe from a single disk failure.

Cost

Since many cloud services are subscription based, we want to design a system that costs less than a year's worth of cloud services. This budget would allow for a cost-effective initial implementation, and then later leaves room for any potential upgrades and maintenance that might be necessary. As an example, consider that storage in the cloud, has considerably high costs for more than 1TB of data. To store that amount of data, Google Drive charges $9.99/month; the cost increases to $99.99/month for 10 terabytes. Contrast that cost to that of a 6TB mirror RAID configured Network Attached Storage (NAS) box with space for expansion, which costs approximately $900. Though the upfront cost seems high, it costs less than 10 terabytes of storage over a year from Google Drive.

Operations

The system administrator is key to the deployment and maintenance of any system. Since home administrators are unlikely to be able to administer the system full-time, we want our personal home cloud to have a relatively simple design that can be quickly and easily supported by the administrator. As part of that, we want to allow remote access to the entire system.

Components

A personal home cloud consists of many components, each of which provides functionality that is fundamentally necessary to the system. Each component has multiple possible solutions, each of which has its own advantages and disadvantages. Here we analyze the characteristics of each possible solution, so that we can design a personal home cloud that best meets our goals.

Static Endpoint

The typical user would not want to remember an IP address, especially one that changes randomly, to access their desired service. There are two options to achieve a static endpoint: purchasing a static external IP address from an Internet service provider (ISP) or having a domain that redirects to the current external IP address. Purchasing an external IP address is most likely not the best option since the user will still have to remember the that address and, additionally, not all ISPs allow residential customers to obtain a static external IP address anyway. Thus the better solution is to have a domain that redirects to the current external IP address. Below, we consider three options for a domain that will redirect: WordPress.com, Tumblr, and Firebase. Our evaluation of each shows that Firebase is the best option.

Wordpress: A common web hosting option is WordPress.com, which offers free domain names under wordpress.com. Obtaining one of these domains is easy: all that is required is an email to create a WordPress account and then choosing an available domain name. On wordpress, free domains have some limitations to customizations, which poses an issue when trying to create a purely redirect site. For instance, the free tier does not allow access to the html code of the website. Updating the website is instead limited to a user interface that allows theme changes and changes to specific elements on the page; this policy disallows redirection. WordPress.com also has paid domains, but a free domain for a home cloud is ideal as once it is acquired it cannot be lost, and paid domains are reclaimed once payment lapses.

Tumblr: Another web hosting option is the popular blog site Tumblr. Creating a Tumblr account is also simple and straightforward, and the domain is under tumblr.com. Tumblr, unlike WordPress.com, allows the user to access the raw html. Not all html tags are permitted, however, and one of the prohibited ones is the meta tag that allows for redirects. (Interestingly, the code blocking meta redirect tag seems to be buggy and work sometimes.) Another option is to use JavaScript to redirect, which is both allowed and works.

Firebase: Firebase, which is a Google product, seems to be the best solution. Firebase offers various backend services targeted for a mobile platform, which includes web hosting and authentication. The hosting Firebase offers is open to all modifications and even uses Node.js, which means it can easily be scripted to deploy a new website whenever the external IP address changes. Thus, Firebase meets all the requirements for hosting a redirect domain. Additionally, the documentation to get started and deploy websites is simple and should be extremely fast to use.

Authentication/Authorization

Authentication and authorization, where the system both authenticates that the user is who they say they are and authorizes access for that user, can be tricky to get right. Below, we evaluate three possible solutions for this component and determine that we require Firebase authentication with username and passwords to achieve both authentication and authorization.

Single Sign On: Single Sign On (SSO), which is gaining popularity. Large companies have created their own versions of such an environment---some even provide SSO capabilities to others, but such services are not free and are intended for enterprises. Creating one's own SSO environment is complex and any bugs might lead to security holes.

Username and Password: Using username and passwords, this standard method provides authentication as well as authorization. To guarantee authorization throughout the system, we could require signing in at every new access to a machine.

Firebase: Using Google's Firebase authentication could be the safest and easiest option. Firebase allows various sign-in platforms to be combined to a single user and is advertised to be created by the same team that developed Google Sign-in.

Storage

A personal home cloud system needs to provide support for the storage of data. We would like that data to be accessible across the network and encrypted. We also want some reliability guarantees. In general, we are not as concerned about performance since the limiting factor in the system will be the upload speed provided by the ISP, which is typically much less than that supported by a hard drive. In general though, the more money invested in the hard drives, the better performance and reliability can be obtained if more money is put into the hard drives selected. Below we evaluate two options Linux disk support and consumer Network Attached Storage.

Linux disk support: Linux provides software for data encryption, accessing a disk over the network, and RAID configuration. For encryption, it provides packages that allow for encrypted volumes or entire disk encryption. The best option for security would be to encrypt the entire drive and then encrypt each volume as well. This way the disk will be encrypted if it ever becomes offline and each volume could also be encrypted and access gated while online. This type of encryption can be sped up if there is hardware support for the type of encryption being used between the hard drive bus and CPU. Hardware-based encryption can also be built in the hard drive which can perform entire disk encryption which no degradation in performance.

For network access, Linux also has packages to remotely mount drives and configure RAID. Mounting drives remotely is simple and a wide variety of protocols can be used. RAID configuration, on the other hand, is not straightforward and requires extensive setup and maintenance.

NAS solutions: Home solution products such as Network Attached Storage (NAS) that promise quick and easy setup and support RAID configurations have been increasingly common. NAS solutions from Synology provides basic security configurations such as https, users and passwords, and VPN. This solution provides many of our desired characteristics in one package with setup support.

Administrator Support

To simplify the work required of the system administrator, we want to provide remote access and unique machine names. Remote access is typically provided through a VPN, while a Domain Name Service (DNS) is used to provide unique names to machines. Below, we evaluate a few options for VPNs and DNS servers. In the end, we choose OpenVPN and a router supporting DNS, respectively.

VPN options: There are a few VPN options, Point-To-Point Tunneling Protocol (PPTP) and OpenVPN being the most popular among them. PPTP, which is considered lightweight in comparison to OpenVPN, is also considered obsolete because its security measures are currently broken and so it is deemed insecure. OpenVPN provides security, which is a desired characteristic of our system, using a combination of generated public keys and a user password. It provides end to end encryption and has many other customizable aspects. This solution easily allows an admin to securely connect to the home network and even maintain the system while away.

DNS server options: To ease the administrator’s task through separation of concerns, we want the DNS server to be a standalone machine. Therefore, we consider DNS software for the Raspberry Pi, which is cost effective, and that on the router itself, which is already necessary equipment.

When setting up a DNS server on a Raspberry Pi there are various potential options. For instance, BIND9 and DNSmasq both provide similar features but require different levels of administration. BIND9, Berkeley Internet Name Domain, offers a DHCP service as well as many other networking tools. BIND9, though, is complex: it contains unnecessary features and requires extensive technical knowledge to configure and use. DNSmasq, on the other hand, could be considered a lighter weight BIND9 since it provides similar features but with less fine-grained control. For instance, simply installing DNSmasq will start as a DNS server, thus it is easier to use and requires less technical knowledge.

The easiest option, however, would be to select a router with DNS capabilities. Most modern routers allow for specified WAN DNS endpoints and LAN DNS support. By simply supplying a domain name, the router then can use client names instead of IP addresses when referencing machines and routing packets. The administrator can also implement what are possibly the most common security measures, subnets and routing rules that isolate machines from one another and control network flow directly in the router.

Distributed Computing

Beyond cloud storage, cloud services also provide virtual machines and cloud computation. Access to virtual machines will require direct access and cannot be protected by a proxy machine. To accomplish this a port will need to be assigned to each virtual machine and direct access would be allowed through SSH. This direct access is not a security concern since it is a virtual machine and completely isolated from the host machine. If the host machine isn't powerful enough, there might be contention for resources, but if such were the case, the virtual machines should not be provided.

Solution

Here we recommend components to build a personal home cloud based on our desired goal of a flexible system that guarantees privacy. We acknowledge, though, that it is certainly possible that other designs are preferable if the desired goals are even only slightly changed.

For the static endpoint, we recommend Firebase hosting, since it can be updated by a script whenever the external IP changes and is only intended to handle redirects. Firebase is a Google product, so Google is providing guarantees around security, performance, and reliability. For the administrator, setup should be relatively easy, and for the budget, there is no cost associated with this choice since Firebase's free tier is all that is necessary.

We would also recommend using Firebase's user authentication service. Authorization would then be handled on the "per interaction" level where any new machine access would also require a user and password. Hosting this website on a Raspberry Pi allows for a proxy machine that first authenticates before redirecting to the storage, which means that only authenticated users can access beyond the Raspberry Pi. Comparing this choice to the desired characteristics of our system, we see that it checks all the boxes, with the only cost being a Raspberry Pi. Additionally, Firebase's authentication service helps protect against malicious attacks by tracking requests and throttling/blocking IPs automatically.

For storage, we recommend a consumer NAS which users would access with another set of passwords. Using a consumer NAS will greatly simplify the setup process and provide easy to use RAID configurations. Finally, a consumer NAS aids the administrator in ensuring security by providing tutorials and extra tools.

We recommend using OpenVPN to provide remote access to the network for the administrators. Both OpenVPN and a local DNS should be hosted on the router, and a routing table should ensure isolation.

Finally, we recommend a single machine to host any virtual machines; this choice helps the administrator with organization and simplifies management of the system. Then SSH would be the default connection to virtual machines and security can be provided by using routing rules to isolate the connections to the specified virtual machine. Performance, cost, and reliability all depends on the machine hosting the virtual machines.

Limitations

No matter how carefully we design a system there were always be inherent limitations. Our recommended system in limited in that it cannot support a large number of concurrent users. Bound by the upload speed provided by the ISP, it is possible that even a single user could bottleneck the entire system. Our system could potentially throttle the speeds allowed for each user, but that would lower the overall quality of service to the user.The availability of the system is also tied to the ISP. If the ISP experiences an outage, our system does not provide a backup network. Any outages experienced by the power grid will also reduce the availability of the system.

In our system, the user may feel frustrated by the required multiple logins, but minor annoyance comes at the cost of creating a more secure system. Finally, the proposed design requires technical knowledge of each component to ensure the integrity of the system.

Other Solutions

Our recommended system is not the simplest, since we chose to balance simplicity and privacy. Since many of the discussed design components are already services provided by companies, it is possible to build a simpler system entirely based on those products. ASUS, for example, provides a Dynamic Domain Name System service with all its routers that allows owners to obtain a registered domain name. The service is completely free and also removes the need to have a script update if the external IP address changes. ASUS also provides various mobile applications to interface a home network externally, thus creating a relatively connected network by simply enabling features and downloading a few applications. The caveat for these features is the lack of privacy, since when using this service, ASUS could be snooping on the connection. End-to-end encryption could help solve such concerns, but it doesn’t stop ASUS from monitoring traffic usage.

Conclusion

The balance between providing security and simplicity is extremely difficult as providing security complicates the environment. We can ensure security in certain pieces of the design by relying on software by reputable companies, such as Google’s Firebase, for those components. We eliminate the risk of security holes between different components of the system by gating each component individually with a username and password.

Fortunately, by creating this examination of characteristics and components necessary for a personal home cloud, we can easily conceptualize a design that would be suitable for any desired set of specific needs in a personal home cloud system.