Global Information Assurance Certification Paper

ATTACKS FROM WITHIN

A LOOK AT SECURITY CONCERNS FOR ASPS

BY TYSON KOPCZYNSKI

Forward

.NET as described by Microsoft, *"extends the ideas of both the Internet and the operating system by making the Internet itself the basis of a new operating system. (1)"* Microsoft goes further to state that in the near future almost every aspect of computing will be across the Internet. Data storage and software usage will become a subscription-based service. The user will "live" on the Internet. It will include everything a user will ever need to use or access. .Net will take away many of the artificial restraints of hardware now placed on modern day computing. In short, software is no longer bound within some form of media to be installed on a machine. Instead, it is viewed as a service much like cable, dial tone, and gas. The software is also delivered much like the afore mentioned services, via a vast distributed network.

Many things are good about this future. For example, I no longer have to download a patch every time a developer decides to fix some problems in their code. Not to mention this will make it extremely hard to pirate software. Plus, my data and applications are accessible wherever I am in the world, provided I have an Internet connection. In spite of this, the change from a decentralized system to a centralized system is not a path free of any downfalls. There are some major security and privacy concerns raised by this proposed future. The fact that this will all reside on a distributed network (a.k.a. the Internet) should send shivers down anyone's spine. Your data and software that you use is not yours anymore. Everyone's data and software will reside in a data center in which one will have to pay for access to it. With Microsoft not the only entity pushing to make this happen, scores of companies have grown from the VC (Venture Capitalists) gardens wanting to push applications over the Internet. Known as ASPs (Application Service Provider), they are gearing themselves to fill the role of hosting Internet enabled applications. Currently the ASP industry is only about 100 million dollars a year; however, this is projected to grow to a 4 billion dollar industry by year 2004 (7), and within many ASPs, not enough is being done to examine the possible risks to both security and privacy. Nor is much being done to remove those risks. Frankly, there are a lot of holes that are being left open in the process of making this new concept work. In fact, fewer then 25 percent of current ASPs can pass a simple 16-question security survey developed by the GartnerGroup (A third party technology research and advisory company) (2). It is the goal of this paper to try and state some of the possible problems raised within an ASP like environment.

Description Of An ASP

In order to first understand the security concerns derived from an ASP, a person must first understand what an ASP is. According to the Webopedia definition, "ASPs are third-party entities that manage and distribute software-based services and solutions to customers across a wide area network from a central data center (3)." The level at which the software-based services are distributed varies widely from just simple software outsourcing to whole IT shops being outsourced. In the case of whole IT shops being outsourced, the ASP would then be called an

AIP (Application Infrastructure Provider). With AIPs, every aspect of an IT shop would be housed in a central data center. This includes corporate data, applications, and user information. Regardless the main function of an ASP or AIP is the delivery of applications over the network. The delivery technology for this can vary from SCO Tarantella, Java Based Clients, WTS (Windows Terminal Server), Citrix MetaFrame, and New Moon's Liftoff. This paper will discuss delivery technologies that are based mostly only on NT Servers using WTS, MetaFrame and Liftoff as examples, and the possible risks associated with those and the hosting of third party applications.

Security Within An ASP

As with any IT shop, an ASP is also faced with the same challenges with securing their networks. They have to deal with both internal and external attacks, take the same steps in determining sensitivity of data, and even determine a security policy for their physical sites and so on. So, in many aspects an ASP would have to meet very similar security requirements as found within a normal corporate network. However, there are some real differences between the ASP's network and that of a corporation. The ASP's network houses other companies including itself. In dealing with each company, the ASP would have to develop different security policies based on that company's access needs. An example is when a company's application needs access to specialized services on the Internet. Another example is if the company needs remote administrative access to a database stored on a server within the ASP's data center. Now complied with this, is the fact that an ASP's network is possibly housing many different companies. It is the job of the ASP to keep those companies separate in such a manner that each company has no indication of the other (4). This is termed as confidentiality.

Confidentiality is one of the main requirements of an ASP. Confidentiality, as it applies to data, is one of the largest concerns right now for the ASP market (5). In most cases, the concern is in dealing with transfers or transactions of data. The movement of data can occur not only on the internal network of an ASP, but also to a client running an application, and data possibly being moved between a network connection to another company's internal network. In short, some level of encryption should be applied to all transfers and transactions. This should be applied regardless of where the flow of data has originated or is ending up.

With so much resources and emphasis being placed on keeping data safe from prying eyes. The ASP industry is forgetting what differences will be brought in the change from a non-centralized system to a centralized system. If anything, the idea of a centralized source for data storage and computing is not new. For years, this has been the realm of giant mainframes with scores of users connected to a single system. So what does this mean? Well in the larger picture, system administrators will now have to place more of an emphasis on security measures from an inside to outside basis instead of an outside to inside basis. It is true that both cases will still have to be dealt with; however, server side or host level security plays more of necessity when placed in an ASP model. With most ASP technologies, the user basically should be treated as if they are sitting in front of a server running an application. Everything that a user does is in fact happening on the server, not on their machine. This is because the application code is executing local to the server, and then by some "magic" the user sees the results on their machine.

What if a user was able to destroy the environment in which its application was executed? An easy way to do this would be to cause the application to crash. If this crash was harsh enough to put the server in a failed state somehow, then all the other users on this server are effectively dead in the water too. If the server OS (operating system) were similar to Unix an easy solution for protecting the server would be to place the user in his own shell environment, this is termed as Sandboxing. By Sandboxing, you have placed the user and all the processes that user is using in an isolated environment; an environment that a user can trash without affecting anyone else using the same server. If the ASP, however, were using Windows as the OS then Sandboxing would not be possible, because in Windows it is a shared shell.

Now The Applications

It is an admirable task for any ASP willing to take on the challenges of hosting any third party applications. Anytime one takes a third party application into a production environment they are doing so at your own risk by trusting that the developer has done both an excellent and ethical job in the engineering of the application. When the application is handed over to the ASP for placement into a data center for hosting, the ASP is basically going on that same trust. There is no way for the ASP to go through any application and test for every single possible problem or back door short of having the source code. Most third party developers would not be willing to give up such an asset.

As it is today, most of the applications are not structured for distributed computing. They can be haphazard with memory, single instance, or keep program preferences globally not uniquely for each user. Some applications even require administrative rights on a machine to run. All this is actually fine if you are only planning to install the application on a single desktop. The following are some issues that an ASP would have to take into account with an application when trying to deploy it in a Windows environment.

Buffer Overflows and Such

Buffer overflows can be a sign of bad programming. This is also a great way to hack an OS and get it do what you want. There can also be other flaws in the application itself that allow a hacker to compromise information security within an ASP. This should be one of the main concerns of any ASP that is planning on hosting third party applications. Being that it is a third party developer, the ASP has no way ensuring the quality of development for an application. Plus developers seem to have the mentality to allow most of the QA (Quality Assurance) work to be done by hackers. Then, when a bug is found, the developer scrambles to fix the problems **(6)**.

Backdoors

Although, ethically a developer should not place any backdoors into an application, the possibility is there, for developers to slip functionality to gain elevated rights on a server. Another aspect of a backdoor is if a developer were to insert malicious code within the application aimed at destroying the server that is hosting it. A possible reason that a developer would do this is if they were to find out that the same server was hosting another competitor in their software market.

Internet Interfaces

Most of the applications that are being developed now currently have built in Internet interfaces. For example, a web publishing application will most likely have a FTP (File Transfer Protocol) interface to upload sites to a web server. Even if the application does not have the interface built in, it still is possible with most ASP technologies to call child applications from an application. So it is then possible for an application to call IE (Internet Explorer) or some other application that allows access to the Internet. The problem with an application having Internet access is the user is then using the ASP's Internet connection to "Surf the Web". This could then have the potential of bringing about all types of possible legal problems for an ASP, because not all users use the Internet for legal activities. Other potential problems would stem if a user were to go to a site that had malicious code on it. Unless the server hosting the application was properly protected then it would be possible for the user to destroy that server.

Macro/Programming Interfaces

Probably one of the best examples of problematic macro and programming Interfaces is found Microsoft Office. The macro abilities within Office have been a headache for systems administrators for a long time. Macro and Programming Interfaces simply allow too much access to systems functionality. So any hacker with some knowledge of programming could potentially write a nasty piece of code, and then run it while using a hosted application. Since the code is executing through the hosted application, the code is actually being executed on the server that is hosting that application.

Windows Policies and Non-Standard API's

Windows has lots of different policies that you can implement in order to secure a system. The explorer policy NoNetHood (used to hide network neighborhood) is an example of one. In fact you can also secure a system by even restricting access to certain dll's, for example the dll adui.dll (UI for viewing file system permissions). However, if the developer does not us common Windows standards when developing an application then attempts to lock down an application server through these methods will fail. For example, it is very easy for a developer to write his or her own API for viewing the file permissions on NTFS.

Conclusion

Many companies are touting Internet delivered applications as the future, and as time progresses the choices to how someone gets their applications or stores their data will start to dwindle. Before people will adopt this new way of computing ASP companies will have to work on not only ensuring that security placed properly around these systems. They will also need to convince people that these systems are truly protected. The problems that were outlined in this paper are pretty much all correctable. In most cases, the reason they are such problems right now is because applications, operating systems, and even networks were never designed to handle the aspects of what an ASP is trying to achieve. Now the true question in this case is whether or not ASPs will take heed to these facts and start down the direction of deploying a secure system in the beginning or will it take a severe breach in information security before they will take notice.

References

(1) Microsoft Corporation. "What .NET Means for Users and Developers." 2001. URL: http://www.microsoft.com/net/developer/developers.asp (01/16/2001).

(2) Cone, Edward. "ASP Security: Priority No. 1." 07/11/2000. URL: http://www.zdnet.com/intweek/stories/news/0%2C4164%2C2601018%2C00.html (01/16/2001).

(3) http://webopedia.internet.com/ (01/17/2001).

(4) JAWS Technologies, Inc.. "Secure ASPs." 2001. URL: http://www.aspinsights.com/docs/artlink.asp?storyid=1082228749 (01/17/2001).

(5) ASP Industry Consortium. "Security, Uptime Remain Primary Concerns of ASP Customers." 10/24/2000. URL: http://www.allaboutasp.org/pr-24oct00.cfm (01/17/2001).

(6) Edwards, Mark. "Application Service Providers: Are They Sitting Ducks?." 04/06/2001. URL: http://www.ntsecurity.net/Articles/Index.cfm?ArticleID=8529 (01/16/2001)

(7) Anderson, Christa. "A Broader and Simpler ASP Market." 03/22/2001. URL: http://www.win2000mag.com/Articles/Index.cfm?ArticleID=8425 (01/18/2001)