Designing Web Environment Security Solutions

John P. Pironti, CISA, CISSP
IP Architects, LLC.
Version 2.0 – May 1, 2001

The world of electronic business has evolved from "brochureware" web sites to fully dynamic and interactive user experiences. Information ranging from bank account data to frequent flyer account summaries is now accessible via the Internet. An enhanced level of risk accompanies this new level of access. Security has become an essential aspect of web environment design. The risk of data being accessed and manipulated by unauthorized individuals has many implications. These implications can range from a loss of productivity to correct the misuse, to significant public relations implications in order to regain customer confidence in using the affected web environments.

There are certain fundamental precautions that can be included in the security design of a web environment to help reduce the risk of malicious activities. It is important to remember that it will be impossible to prevent all types of malicious activities. If an attacker is determined, they will eventually find a way to compromise the web environment. It is also important to gauge the cost of the security solution that you employ in a web environment against the value of the data that you are trying to protect. It is not financially responsible to invest $500,000 in securing a web environment when the cost of this data being compromised is only valued at $250,000.

When determining the security requirements for a web environment it is important to review the physical, network, and application elements involved. Each of these elements has its own unique risks, and unique protective measures that should be put in place to manage these risks. These measures should be described in security policy and procedure documents. These documents, when accompanied by strict enforcement, can potentially have as much impact on the maintaining the integrity and security of a web environment as the technology components.

Physical Security

The physical location and placement of the security components included in a web environment can have significant security implications. It is important to limit physical access to the equipment to authorized personnel only. Although this may seem like an obvious consideration, it is sometimes easier said than done. In the 24x7x365 world of the Internet it is easy to lose track of who has physical access to what and why. While posting armed guards in front of your equipment is probably not required, it is important to keep detailed logs of who is handling your equipment and what they are doing. This is important for both a chain of custody in a security audit, as well as extremely helpful in troubleshooting these environments when system failures occur.

In order to protect your web environments from being physically altered you should consider housing them in a quarantined data center environment with access control mechanisms in place. These mechanisms should include photographic identification badges with proximity strips that can be read from proximity readers at access points. It is important to be limit access to authorized personnel only, but it is important not to put so many restrictive measures in place that your staff feels as though they are working in a prison environment. If your employees understand the reasons for the measures you are taking and agree with them they will tend to work hard to reduce the human factors that can compromise a security solution. If they feel as

though they are being watched and investigated at all times they may actually become an attacker as an act of defiance against you. The most effective way to motivate an employee to be an ally instead of an enemy is to educate them about the risks that you are trying to mitigate, and inform them as to how essential they are in this process.

There are three basic elements that are normally used in physical security authentication schemes for individuals to be granted access to a secure physical environment. These elements are "what they have", "who they are", and "what they know". It is important to consider these elements when designing your web environment's physical security as it relates to personnel access. Photo identification cards with proximity strips built in are a simple and effective method of controlling and auditing access to restricted environments. One common mistake often made when using these cards is the lack of a pin code element. Without the pin code only two of the three critical elements have been met. The photographic identification card with the proximity strip can be misplaced or stolen, but when accompanied with a pin code for access the intruder will have to gain intimate knowledge of the authorized individual from whom they obtained the card. These pin codes should be changed every thirty to sixty days and be a minimum of six characters in length.

An easy an inexpensive way to ensure a higher level of physical security of a web environment is to put it under constant video surveillance. This video surveillance should be encoded with date and time information, and all physical contact with the media which the video is stored on should be documented in order to be admissible in any judicial proceedings. By advertising the fact that video surveillance is in place you can potentially deter an intruder from attempting to access or cause physical damage to a critical system. It is usually prudent to archive this surveillance video for a period of at least one year. This will ensure that you can thoroughly research inappropriate activities, as well as prove that these activities are not part of the normal working procedures of the facility.

Network Security

In order to create a plan for network security it is important to first understand the network elements that that you are working with. These include the type of devices and communication protocols, which are required in your web environment. There are many protocols and components of protocols that are currently used in computing environments. In web environments the TCP/IP protocol is usually the primary protocol that is used.

The TCP/IP protocol is a packet-based protocol that consists of 1,024 defined logical ports and 65,536 logical ports in total. The defined ports are address assignments that are specific to certain types of applications, which are used to communicate with systems that use this protocol. Some examples of commonly used defined ports include port 80: HTTP, port 443: SSL, and Port 21: FTP.

Using defined ports within web environments allow you to gain greater control of your web environment. Third party security vendors are aware of these ports and applications, and their products will usually be able to work with them automatically. An example of this would

be the ability to block specific data by application type instead of port numbers within a configuration program for a filtering device such as a packet filtering router.

It is important to plan your protocol requirements within your web environment carefully. While certain protocols may allow for more efficient data transport and easier application development, it is always important to remember the balance that has to be understood when planning for security. How secure do you want to be, versus how much you want to pay, and how much performance you would like to achieve. When dealing with web environments we can usually simplify this. In most cases the only ports that need to be opened to the Internet facing aspects of the web environment are port 80 (HTTP), and port 443 (SSL). This does not mean that you will not have other ports opened to local LAN transport between servers within in the environment. By opening only these two ports to the Internet facing interfaces you can greatly reduce your network risks and also focus your attention to these ports and the security risks involved with them.

When dealing with a web environment you will use a router as your interface device to the Internet. This router can be characterized as a traffic cop and guide for Internet traffic. It sends and receives traffic to hosts along the Internet based on IP headers that are part of each data packet. This IP header includes essential data that is included within the packet. Key data includes; the source IP address, the destination IP address, the size of the packet, and the port number the packet is attempting to access.

The Internet facing router usually has certain security features that can help enhance the security of the web environment. The router also has the ability to filter traffic based on the information included in the IP header. These filters are known as access control lists. It is important to setup these filters in a DENY ALL and then set the PERMIT function to allow traffic to the web servers and specific ports on the web server. This will ensure that the router logs all packets and sessions that are destined for the web server and automatically block all ports and protocols that you do not want to access the web server. It is also important to disable the routing of private IP (IETF RFC 1918) addresses, which should only be used in local area networks.

There are some other security precautions that should be taken when configuring the Internet facing router to enhance the security of the web environment. These include setting the DENY ALL function to block all packets and ports above 1024. Cisco routers will only deny protocols up to 1024 if not explicitly instructed to filter higher ports. An attacker will often use a port scanner such as an ISS system scanner or NMAP to scan your router to see what kind of traffic and which ports it will respond to. If an attacker is able to exploit an undefined high port they may go unnoticed because you are not specifically monitoring activity on these ports.

When implementing up your web environment it is important to establish a segregated subnet from your corporate network for the environment. This will allow you to mandate specific security policies and configurations that will only effect your web environment, and not your entire network. The added security will limit an attacker's ability to gain access to your corporate resources if an infiltration occurs.

In most cases you will have multiple layers of servers included in your web environment. A three-tier architecture that includes web servers, application servers, and database servers is now common (figure 1.0). In this design, the only element that has to be accessible from the Internet is the web server. This is an ideal situation in which to use layer two switches to facilitate communications between the servers.

Layer two switches not only enable high-speed communication between servers but also enable a higher level of security than router communication. They do this by communicating with the servers through the media access control (MAC) address that is uniquely assigned to each server network interface card instead of only using the IP address that is assigned to the server. IP addresses can be easily spoofed to fool a server into believing that the traffic it is receiving is coming from a trusted source. MAC addresses can be spoofed as well. This is much harder since the attacker must first determine the MAC address of the interface on the server they are originating the traffic from, and then obtain the MAC address of the interface they are attempting to compromise.

Layer two switches allow for virtual local area network (VLAN) to be put in place for the networking aspects of an environment (figure 1.0). In this case you can use the VLAN to create separate communication segments. When communicating with an Internet facing router you must utilize a public IP address in order to properly advertise your server to the Internet. This is not a requirement for equipment that is not exposed to the Internet.

The non-Internet facing equipment should communicate with each other through separate VLAN segments (figure 1.1) and use IETF RFC 1918 private IP addresses. This will allow sensitive traffic to be transmitted between servers without being exposed to the Internet where it potentially could be captured by an attacker. This traffic should be encrypted if possible, but even without the encryption this technique will increase the security of this traffic.

Virtual Private Networks for Data Transport

When you use a web-hosting provider to host your web environment you will have a new range of network security issues that need to be addressed. One of the most important is the security of the data that is being transmitted between your environment and the web hosting provider's environment. An easy and efficient way to secure this traffic is to establish an encrypted Virtual Private Network (VPN) between your network environment and the provider's data center (figure 2.0 – 2.1).

Virtual private networks allow data to traverse the public networks (i.e. Internet as a transport mechanism while maintaining the security of private connectivity). This is accomplished by using encryption and authentication technologies, which ensure the integrity of the data while it is traveling over the public network. VPNs allow you to take advantage of the availability, reliability, and scalability of the Internet without having to incur to the cost of building out your own private network capabilities.

An IPsec compliant (IETF RFC 2401) VPN that utilizes digital certificates for authentication will offer you a high degree of security, data integrity, and interoperability with

other data providers. IPsec offers link integrity between two network access points using the Internet as the transportation medium. It does this by using strong encryption (168bit 3DES) with digital certificates as the authentication mechanism. Each packet is authenticated and verified to make sure that it was not changed in any way during transmission.

IPsec VPNs also include mechanisms that protect users of the VPN from a "replay attack". In a replay attack the attacker will attempt to capture packets in transit using a network sniffer. The attacker will then retransmit these packets back to the host at a later time to simulate the original transaction. This becomes extremely crucial when you are transmitting database updates between hosts. If an attacker were able to retransmit a series of transactions back to the host it could invalidate the data in the database. This would require a database audit and the need to potentially revoke all transactions since the last database backup occurred.

Application Security

Application security should start at the software design phase. Unfortunately, many software engineers and developers find input from security professionals helpful but obtrusive. The suggestions that security professionals often will make tend to limit the capabilities of the software, and extend the development time required. It is very rare that a software engineer or developer will intentionally design or develop their software with security flaws. The current trend is to produce software at a fantastic pace that does not allow for thorough testing. Software that would traditionally be considered beta code with a limited testing audience is now being used in production environments. This phenomenon becomes blatantly apparent when you consider the number of service packs and patch releases that software manufactures release on a regular basis. The vendors are now using the end user to test their software, and make them aware of the bugs in their code.

This situation presents an interesting challenge when attempting to secure a web environment. While the installing of vendor-released patches is recommended, you must be careful when implementing these patches. Vendors tend to release security patches in two forms. Hot patches, and fully tested patches.

Hot patches are released when a high-risk vulnerability has been discovered and endangers the integrity of the system. These patches tend to remove the vulnerability, but often have not been thoroughly tested for their effects on the operating system or other programs that may be in use.

Fully tested patches take longer to be released, but have a higher level of integrity associated with them. They tend to be released with a number of other patches at the same time, and also have extensive release notes associated with them. These release notes are essentially the test results from lab environments, which can help a user predict how the patch will affect their systems. It is recommended that you wait for a fully tested patch to be released from a vendor if you do not have the resources to establish a test lab to test hot patches prior to installing them in production environments.

The question that has to be asked at this point is "What do I do while I am waiting for the fully tested patch to be released to help counteract my system vulnerability?" The first thing to establish is to what extent does the application's vulnerability affect your system given the operating environment you are running the application in. Many of the vulnerabilities that are discovered only affect specific configurations. If your environment is at risk, you must make sure that you have other application level security precautions in place to help identify when someone is attempting to take advantage of the application's vulnerability to verify the integrity of the systems that are affected.

There are two common methods that can be employed to achieve this goal. Host based intrusion detection systems and file integrity verification tools. These two tools differ in their purpose and use, but together create a significant layer of protection for the systems that they are installed and used on.

A majority of the current host based intrusion detection systems work with a combination of heuristic data, and attack signature matching techniques. In this case, the data will be examined during the packet processing activity. The heuristic approach will watch specific activities on a host, and based on the security policy that is in place will trigger an alarm and potentially countermeasures to repel the attack. An example of an attack that a heuristic system would detect is the acceleration of permissions within a system. If a user is logged into the system with a guest level permission and suddenly has administrator privileges this would be something that the intrusion detection system would be expected to flag as a potential threat. The signature approach works along the same lines as a virus scanner works today. The vendor will provide signatures of attacks on a regular basis and will compare system activity with these signatures.

Another approach to host based intrusion detection is to examine the system and event logs that a system generates for suspicious activities. These systems have the benefit of being less intrusive to both operating system and the applications that are running on the host. These tools still follow the same heuristic and signature matching approaches, but do so after the activity has been captured. When using these systems it important that you enable all of the appropriate logging features within the operating system and applications that the intrusion detection software can analyze. It is also a good idea to have these logs written to a separate host and analyzed on this device. This will reduce the risk of an attacker being able to delete the logs in order to defeat the intrusion detection solution, and hide their activities.

The real key to these tools is the logging that they can offer. The logs of any intrusion detection systems and all system logs should be offloaded to a system outside of the web environment. This will prevent an attacker from deleting or modifying the logs to cover their tracks. These logs can then be used to determine how the system was accessed and what elements were accessed or modified.

File integrity tools will generate one-way hash signatures of files that can be used to detect file tampering or alteration. One-way hash functions convert messages of any length into a fixed-length message digest. This message digest cannot be reversed and is unique to each file

that is hashed. Changes that are made to the files will result in a different hash result after it has been processed.

Once you have a library of message digests from hashed files it is important to store them on read only media (e.g. CD ROM). This way you ensure that the hashes cannot be tampered with in any way. You can then use this library as reference which you can compare future message digests of the same files against. If any of the message digests of the newly hashed files do not match the reference copy you not only know that something has been modified, but you will know the exact file that was modified. It is important to note that file integrity tools work well for files that are not changed often (e.g. system files), but also are not appropriate for files that change on a regular basis (e.g. dynamic transaction databases). It is also important to generate new hash libraries each time new software or content is installed in the environment. This will ensure that the comparisons are always being performed on the most recent version of the production installation.

Honey Pots

Once you have taken all the measures that you can to secure the host, you can take measures to lure attackers away from your host by creating a more attractive target for an attacker. This technique is commonly known as deploying an "electronic honey pot". A honey pot is a tool that is setup to create the facade of an actual host, which is vulnerable to attack. Attackers will tend to look for the easiest attack point in an environment, and a honey pot becomes the most vulnerable point in the environment.

When deploying a honey pot solution it is important to understand what it is you are trying to accomplish with the device. There are different versions of honey pot software currently available. Their functionality can range from merely creating a convincing facade and logging an attacker's activities, to tools that will log the attackers activities, trap them within a sandbox environment, and attempt a trace back an attacker to a source computer. The level of complexity that you choose to deploy will depend on your commitment to maintaining the honey pot itself.

Honey pots present an interesting problem from maintenance and support perspective. Commercially available honey pot software will tend to be fingerprinted in relatively short period of time. This fingerprinting will allow a savvy attacker to understand that they are interfacing with a false device and move on to another device in the network. An attacker can also identify the honey pot if the content on the server is not kept up to date or the software itself is not kept up to date with vendor updates. This means that the honey pot must be included in the content update scheme for the environment, as well as closely watched and monitored. If it is not kept current it will quickly become and expensive element of the environment that enhancing the overall security.

Final Thoughts

The key to any security design or implementation is the understanding of what it is you are trying to protect, and whom are you trying to protect it from. With this understanding you

can ensure that you are deploying your resources appropriately and can justify your activities. There is no one solution that will secure your web environment against every situation that you will encounter. Security designs commonly resemble Lego® sets. There are multiple elements that come together to create a complete security solution. Individually each element serves a purpose, but when put the elements together they create something better. The instruction set that you follow to create the security solution is the security policy. This policy sets the stage for any and all security related activities that will take place within the environment. Once you have followed the instructions and put all of the pieces together you will have a security solution that will meet the requirements of your web environment.

Figure 1.0

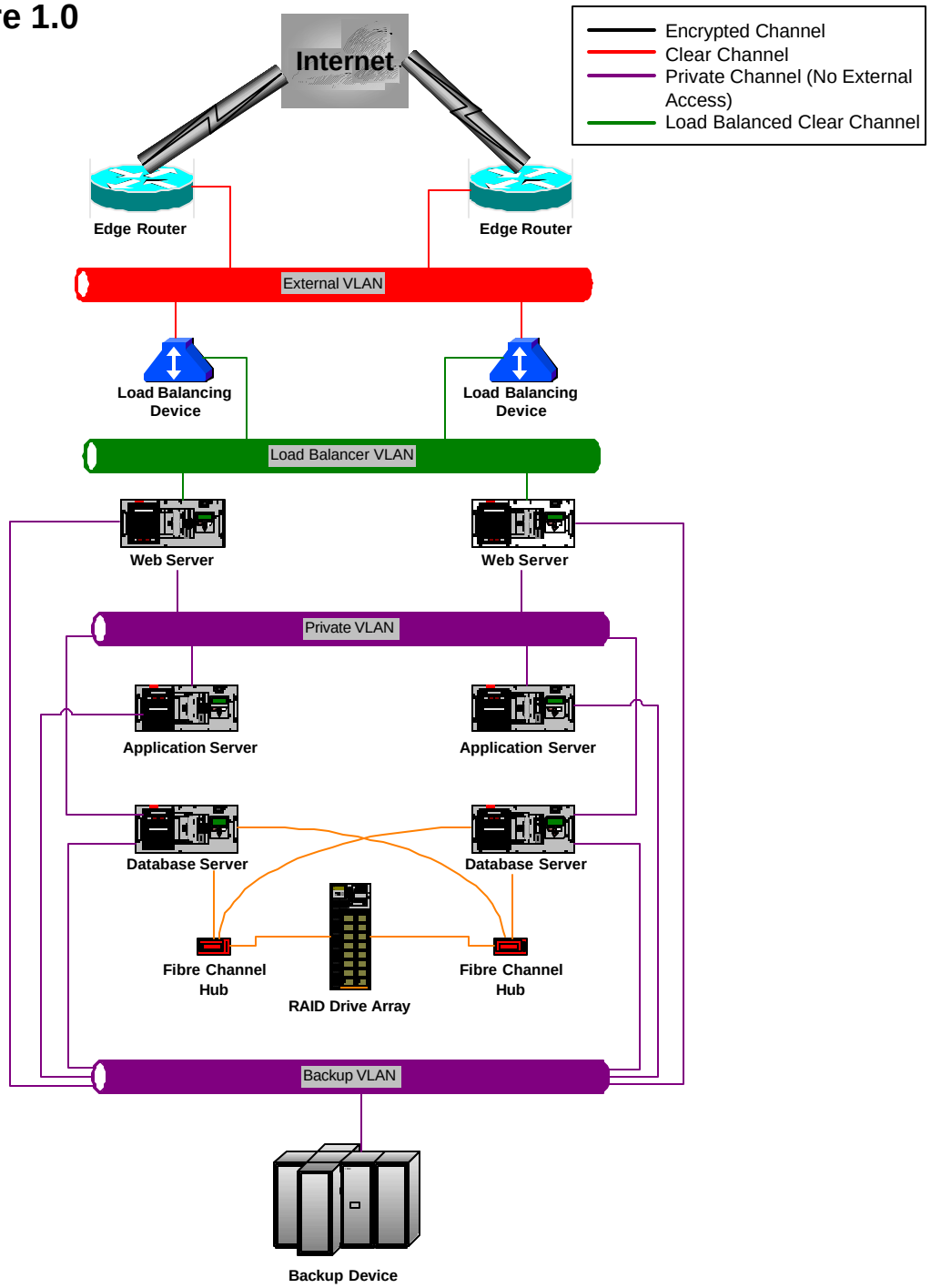

Figure 1.1

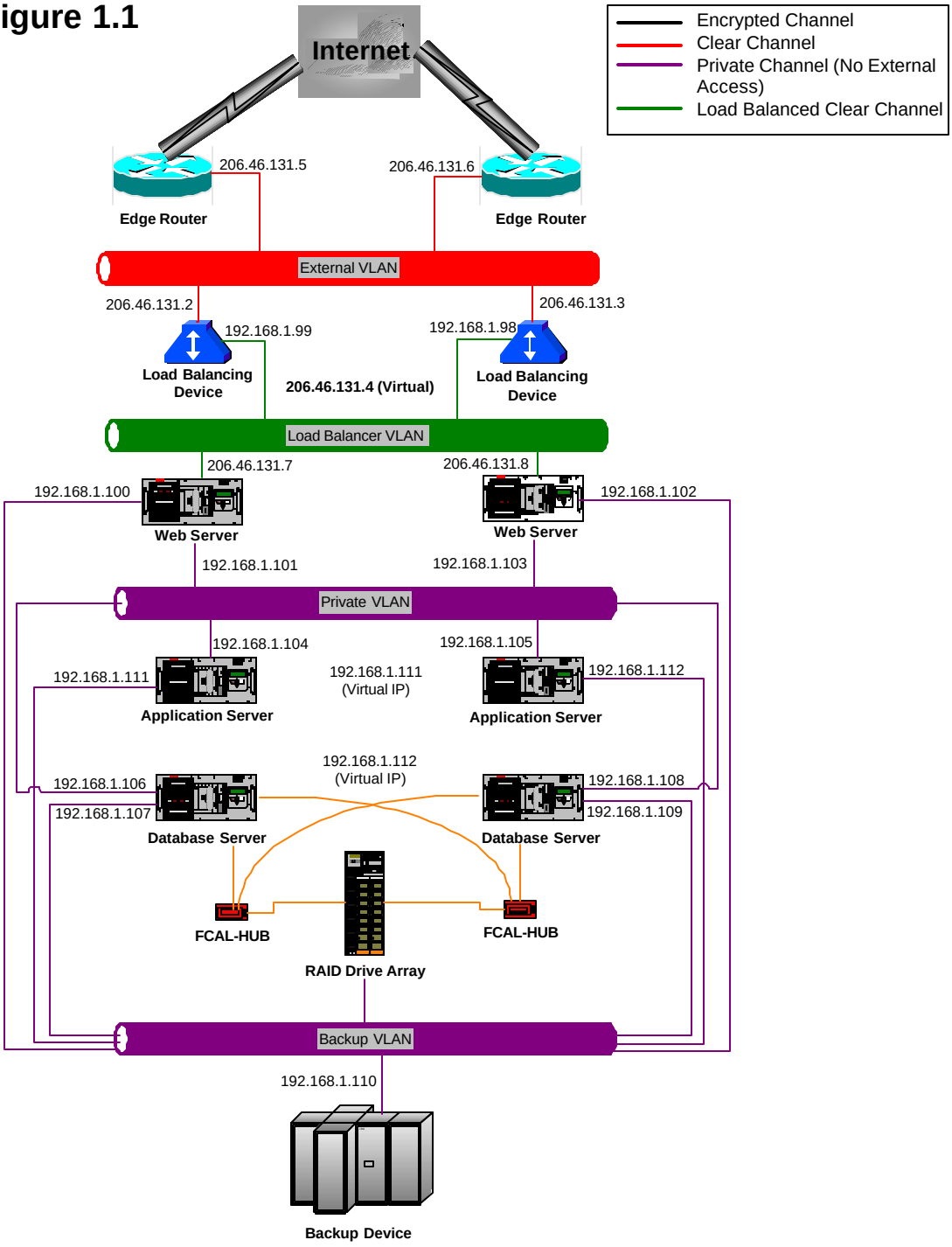

Figure 2.0

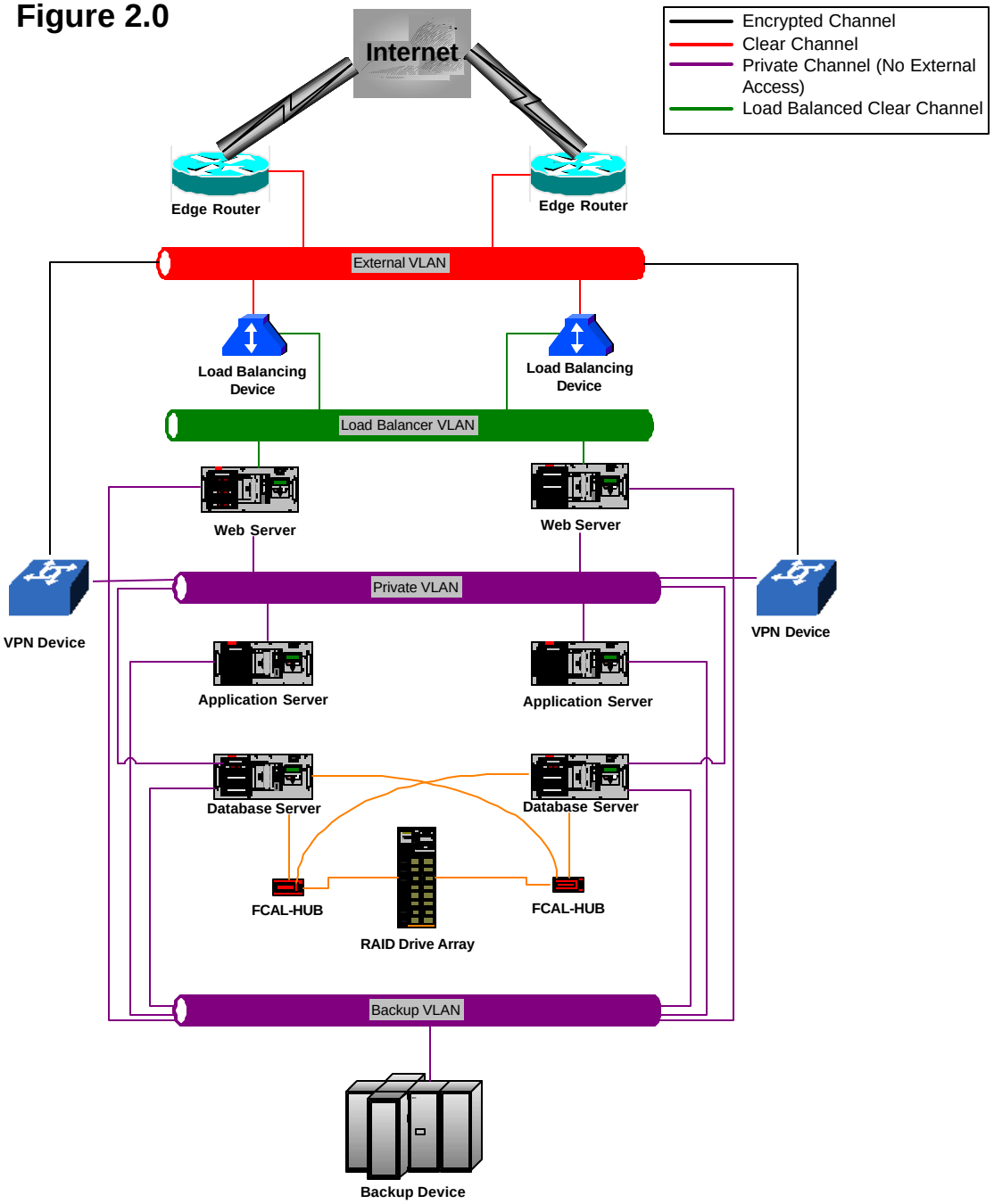

Figure 2.1