An intrusion, in an outsourcing data center, that works in spite of security

Rick Kryger

An intrusion, in an outsourcing data center, that works in spite of security
Rick Kryger
October 16, 2003
GSEC practical version 1.4 – option 2

ABSTRACT

I work in a data center, focusing network connectivity and network security. Solution architecture centers on providing the layered security promoted by the industry in general and SANS in particular. We strive to incorporate the highest level of security and performance permitted by the customer, budget, and availability of stable, suitable technologies. We buy or receive what is agreed fits the solution and implement it in a way that it will perform well. After implementing the environment, the next step is to monitor the individual components of the solution and keep up with required changes to the network components (intrusions, patches, new versions and technologies.) Periodically we, or a third party, with the approval of the customer, will attempt intrusions or perform simple audits of the environment. This approach to security, typically effective and reproducible, combined with a fairly constant and high workload, left me with the solid belief that the networks were secure and that we were doing what was needed to continue the trend. However that changed one day when one customer reported that a break-in had occurred.

After a conference call, it was time to find out: 1 – how; 2 – when; 3 – how often; 4 – what was changed; 5 – figure out how to stop a repeat occurrence and how to prevent future similar break-ins. After discovering the above five items, we could recover from the intrusion and then hopefully publish new firewall rules or utilize other security tools to both recover from and eradicate the vulnerability that led to the intrusion. We eventually did document the “break-in” – it occurred after a support person, working for/at the customer site, re-enabled a default high-level system account with the default password and a user across the country stumbled upon it. We created a secure environment for the customer, monitored the components, kept up to date with changes and technologies, but for not. No matter how secure the architecture, how complete the procedures, or how diligent and skilled the network support team is, nothing short of knowing and analyzing all changes inside and outside of the solution can protect an environment completely. One of the lessons learned because of the intrusion was not only to concentrate on implementing and monitoring the things intended to provide and protect the solution, but also keep in mind that the impossible, such as logging into a secure web site using a high-level login with the default password, may be possible and beyond your control and/or knowledge. So since the intrusion, to attempt the impossible, as time permits, is now not only not a waste of time, but is a new addition to the best-practices list.

BEFORE

This particular customer, hereafter termed “the customer”, has more than twenty sites geographically separated by from several to hundreds of miles. Each site requires non-overlap of information, to the extent that the general plan

is for a separate LDAP domain for each site that is located at their site while the web front-end, application servers, process schedule servers, data base and data warehouse servers are located at a site, the data center that I work for, geographically separated from them all. One entity, here after termed a separate site, acts as the central IT group and is ultimately in charge of satisfying the separate sites and help their local IT group resolve issues. This last site is the one who contracted us to house and to varying degrees support the enterprise, promoting a level of standardization and to reap the benefits of economy of scale. All sites utilize the same ERP solution, but different sites do not necessarily use the same version or have the same options enabled. The data classification for the ERP data would be sensitive but unclassified.

- To combat physical vulnerabilities, within the data center, very few single points of failure exist. The entire data center utilizes an Uninterruptible Power Supply system that includes battery backup and backup electricity provided by diesel generators. The customer's serial connectivity travels across two separate WAN lines terminated at one end in the data center and at the other ends several hundred miles apart. The data center LAN utilizes switch and firewall pairs with dual power supplies. The dual power supplies are connected to two separate Power Distribution Units, PDUs for the rest of this document. Most hosts, within the customer network, have dual power supplies, which are connected to separate PDUs, and dual network interfaces connected to paired switches. Fail-over protocols such as VRRP, HSRP, and multi-path are in-use on firewalls, routers, and hosts. The web service is front-ended by a content switch pair sending connections out to a web farm. Application and process schedule servers are implemented in pairs both for redundancy and load sharing. With this particular customer, the only blatant single point of failure is the database and data warehouse servers, as these entities are not clustered. The database and data warehouse servers are connected to dual everything, but have the vulnerabilities inherent with any single host – those being relative to CPU, RAM, and to a lesser extent mass-storage issues.
- An anti dumpster-diving policy in-place is that after specific documents are no longer needed, the hard copies are shredded before begin discarded to the trash. The specific documents include: Information System Change and/or Service Requests; IP address assignments both routable and non-routable; add network device configurations.
- Network information reconnaissance is further limited because there is no write SNMP community string and the read community string is complex. Also, SNMP traffic is only allowed to flow thru the network to particular monitoring devices.
- To alleviate many incoming e-mail and DNS zone transfer issues, e-mail and DNS access for network egress only. The e-mail access is to send monitoring exceptions out to customer and data center technical support teams via e-mail messages and/or pages. The vast majority of DNS access is inter data center only, no TCP port 53 is allowed into or out of the data center. UDP port 53 is allowed to originate inside the data center

and leave for specific admin hosts for patch download and as a part of the Sendmail process. The hosts with allowed egress, out going, e-mail and DNS access in the data center cannot hold production data and are monitored closely.

- To increase the amount work required for a would-be intruder to gain login access to network devices within the data center encrypted access is allowed to specific addresses only. Network access to network devices is limited to very few IP addresses, all the addresses are owned by the company that work for. SSH is the only allowed shell type access to network devices, with exception of SSH and Voyager on Nokia firewalls. Network device passwords are complex. The most complex hashing, MD5, of passwords available on Cisco gear is in use. Network devices that do not support SSH are only accessible after logging into a device that supports SSH first then accessing the device.
- Unix system administrator privilege is granted thru PKI. NT system administrator privilege (terminal services and Citrix) is granted via domain controllers that are accessible only within the data center.
- Operating system login passwords are verified as complex. Periodically the two system administrator teams, with the approval of the customer and management, run password crackers against their respective security domain authentication providers. This activity is to verify password complexity, in addition to the programmatic checks made when a user changes their password. Any account cracked during the exercise is added to a list without the corresponding password. If any accounts, besides the control accounts that have no real access, are found to have non-complex passwords, or easily guessed passwords, the list is forwarded to the customer so that the user can change their password. This activity is not performed on application level or database accounts.
- File hashes are available in the event of expected compromises. MD5 hashes of files, created when they are released for production use, are stored on non-production hosts. These hashes are compared against hashes of the same file names in the event possible unauthorized modification.
- Access to specific hosts inside the data center by technical support outside the data center is limited. This customer requires extensive, basically 7x24, access to specific hosts and network monitoring output. Virtually no hosts within the data center have addresses that are routable thru the Internet. The hosts that have a requirement for access, from outside the data center, are accessed after a NAT function on perimeter firewalls. The hosts inside the data center requiring access from hosts outside the data center include: application servers; monitoring data presentation hosts; and Citrix servers. Access to the Citrix environment includes published applications as opposed to published desktops. Unix hosts utilize sudo in place of su or allowing users to log on thru the network as root. Almost no unencrypted data is allowed into or out of the data center. A higher level of scrutiny is levied against the hosts that the

customer technical support has direct access to, as these hosts are more likely to become compromised. Although, this particular customer has no documented compromises or unauthorized access to the hosts that are directly accessible from outside the data center by technical support.

- Routing within the data center LAN occurs only via internal firewalls and static routes. Routing at the host level is controlled via host files and static route tables on the hosts. Hosts that have connectivity to multiple tiers have specific rules controlling traffics. As an example, on hosts that have connectivity to both tier 1 (Internet) and tier 2, Internet users' traffics are only allowed on specific interfaces with data center traffics directed, originally by the host and later verified by firewalls, thru separate interfaces via static routes. Without fairly extensive modifications, to multiple entities, hosts within the data center cannot participate in the user experience provided to the public. The modifications are made for a limited number of hosts to allow monitoring of what the bastions provide to Internet users.
- Repeatedly exploited protocols and risky access are not allowed into the data center. As an example, no telnet, ftp, icmp traffics are allowed into the data center. So thankfully, the majority of the Internet problems, Microsoft specific viruses, e-mail and web-centric vulnerabilities (Base64 and Apache exploits), and overflow exploits (Open SSH) do not get into the customer's network.
- Desktop type security issues are avoided in the customer network. No user or desktop type hosts are permanently housed within the customer network. Virus protection software is installed on servers with the virus signature updates provided by central data center administration servers.
- Vulnerabilities inherent with application testing and modification are abstracted thru the use of a dedicated test network. A separate test network exists including separate (non-production): IP address space; content switch; firewall; and hosts.
- Formal procedures help keep production environment, for the most part, stable and un-frequently changing. Service Level Agreements keep the vast majority of changes from ever occurring during production hours. Change Requests and Service Requests must be approved and planned in advance keeping experiments from being tried in production. The Change Request and Service Requests procedure includes requiring approvals by both the customer and the company that I work for – the approval process within the company that I work for includes independent buy-offs from technical and management groups.

The ERP package is used fairly constantly by the sites themselves and has cyclic usage from Internet users. The peak usage is from Internet users and happens three times a year. During non-peak usage periods, the bulk of the traffic comes from relatively few addresses.

The latest major version of the software includes two major changes that have had profound impacts on solution performance and security. The two major changes are:

1 – The addition of the customers' customer self-service. The self-service portion of the solution was implemented via https, or encrypted web pages, and available to all Internet users. The application server tier to the solution has increased load in the new version to support this addition.
2 – The inclusion of technical support access to the solution via the web. The technical support access portion of the solution was implemented via https, or encrypted web pages, and available to all Internet users who pass access controls within the application. In previous versions, support access to the application was via configured sockets defined on the application servers. So support people needed firewall rules implemented to allow access from their IP address to the application servers plus a login password combination. In the new version the https web, port 443, is the only access that needs to be allowed thru the perimeter firewall for the application to function. The application servers still exist, but the web server establishes required connectivity thru tier 2 interfaces.

The site that reported the intrusion was running with the newest major version of the application. The intrusion occurred, thankfully, during a non-peak usage period.

DURING

So one day, while contemplating the beauty of the mathematics involved with encryption, the boss came by and basically said "you, in my office now" as we were about to have a conference call with customer X. At the time this seemed odd as most of the conference calls are planned well in advance. During the call with the customer, a site representative indicated that a particular table was updated with inappropriate information. The only other detail available was the time that the modified table was last updated. The change was so obvious that I was not sure what to think. If the hacker was good, was he or she finished and slamming the door loud enough for the whole neighborhood to hear? Was someone throwing down a gauntlet daring somebody to catch him or her? Could it be a script kiddy using something that they did not understand, given to them by someone else interested in observing the standard method of operation in incident handling at the data center? At any rate it was time to get started analyzing the available information and gathering new information.

- We started gathering network packets currently running across network. I requested and received approval from the customer and my management to start multiple full network traces to and from the hosts providing the solution to end-users. After getting the approval, I went thru documentation to define which hosts were used to provide the software solution, hereafter termed THE HOST LIST. The network system administrators' team to start full network traces including all data flowing thru customer network then used this list.
- I stopped data traffics to and from the data center from unknown Internet IP addresses. I requested and received approval, from the customer and my management, to stop ingress and egress data center

connectivity with the exception of primary customer technical support addresses. After getting the approval: added a rule allowing specific address access to the https service; disabled the rule allowing general access to the https service; then published the new rules. The inherent drop not-allowed traffic, that exists in the perimeter firewalls, stopped new traffic. Finally went thru active connections and dropped any established sessions not running to the small list of primary support addresses.

- Then, I started reformatting the events that were transpiring into notes. The notes included all the usual information including: intrusion details; date and time; my name; rewritten notes from the conference call with the customer; and verifications that the agreed upon changes were implemented and time stamps of when they were completed. Where possible, we included screen snap shots and script outputs redirected to files. This customer has never had to involve law enforcement, but at this point in time, we proceeded as though law enforcement involvement was a possibility. So incident preparation, identification, and containment were complete, now it was time to start down the road to see what was/had transpired so we could eradicate and where necessary recover either manually or via one of the regular backups.
- We verified all running ports were intended and running as intended. Using THE HOST LIST (see Started gathering network packets currently running across network), the system administrator teams, in a detailed manner, listed then verified that all currently running ports were intended and necessary to provide the application to users. In a much quicker manner, the system administrator teams listed then verified that all the current running ports did not seem inappropriate or unusual, on all other hosts within the customer (several sites) network. The latter step to verify that a disconnected compromised host had not gained unauthorized or some vicarious access to the site reporting the intrusion. All running ports were allowed and planned, as expected as firewalls control all network access for this contract. No ephemeral ports were running for excessive times and a quick sample validated that no odd encapsulation or morphing was occurring in the sample.
- A common method for covering tracks and removing intrusion clues used by intruders is to remove logs (or at least copies of logs stored/queued on the local host prior to sending to the syslog server) and reboot host. The system administrator teams verified that none of hosts from THE HOST LIST (see Started gathering network packets currently running across network) has been rebooted at an un-planned time. Monitoring software would have generated an alert had this occurred, but better safe than sorry. At the same time the teams verified no reboots, the teams also verified that no system logs were erased.
- A common method used during intrusion is to escalate user privilege. Similarly a widely used backdoor, used to gain future access, is to

create high privilege accounts for later use by the intruder. System administration teams verified that no new accounts were created. The same teams also verified that privilege escalation for a few days preceding the intrusion was valid: granted to users with a business requirement for access to complete planned and/or normal activities again in compliance with business activities.

- Print jobs are a means used to send recognizance and targeted data to intruder. This customer's network limits print job traffics to specific printer devices. A check of logs indicated no large, unplanned, or unusual jobs were run in the recent past – the mass majority of reporting used by this customer is via an on-line data warehouse and scheduled, off-hours, jobs. No jobs were queued for printing at the time of this analysis.
- Since the test network could be used as an abstraction to send information to the intruder, we looked to see what had transpired in the recent past. No new development or database cloning had started, for the specific site, in the recent past. Separate teams scoured the test network looking for anything that was unusual or unplanned; luckily the test network was testing an entirely separate application. Everything in the test network was verified as intended and in-test by the customer. The traces already implemented would record new traffics to the test network, so it was time to continue the hunt in the production network.
- Verified current running TCP/IP sessions not updating inappropriately or doing anything else unintended. Looked thru all current SQL*Net sessions, the full packet data available since the approved traces were running, and found nothing unusual. The only established sessions outside of the data center were to primary support team desktops. Sessions internal to the data center, again primarily SQL*Net, were running as scheduled.
- System administrator teams validated hosts were not compromised. The teams started on hosts, primarily application servers and Citrix servers, accessible from the Internet not compromised. First the system administrator teams compared all current key files MD5 hashes to hashes stored when the last official modifications were made, no differences were found, so none of the files were modified. Next the teams scanned for root kits, but found nothing, as expected since the hosts have anti-virus applications and scripts running continuously. After the Internet accessible hosts were verified, the system administrator teams verified all other hosts in the same manner as they had the Internet accessible hosts.
- One way to hide unauthorized traffic is to run it thru ports left open to fulfill business requirements, so I verified that no unusual traffic patterns were recently running across ports intentionally left open. I looked for backups or any support sessions that started at unusual times or that ran for inordinately long periods. This proved to be a long-term and daunting task as gigabytes if data flow thru this

customer's network every day – and there were a lot of people eager to hear of any progress. After an initial look within a short time-frame surrounding the intrusion I found no smoking gun, so I decided to forgo finishing the complete analysis of all possibilities to focus on other avenues, that would finish in less time, and eventually return to complete the analysis.

- If the intrusion occurred thru the components implemented to service the application for the customer, then one or more of the components could have been compromised. This customer publishes an ERP application on the Internet via https. Public access to the application is controlled with separate pairs of firewalls. The perimeter firewall performs two functions: 1 – it allows the general public access to the site; 2- it translates the public address to non-routable addresses used within the data center. Next a static route sends the packets to a firewall address unrelated to the NAT address. The firewall then forwards the traffic to a content switch which knows the NAT address as a virtual address that is used within a content rule to service users. Finally the traffic is forwarded to any one of several bastion servers that make up the web server farm. By now the traffic has passed thru five different operating systems. The idea that all of the different platforms were compromised seemed unlikely, but any one of them possibly. The network system administrator team went thru the network components and found no comprises. The other system administration teams had the responsibility to do likewise on other systems.
- We needed a listing of all IP addresses that had accessed the IP addresses associated with the application within the two weeks preceding the intrusion. So I went thru perimeter firewall logs and created a list, that included all hosts outside of the data center, that accessed THE HOST LIST (see Started gathering network packets currently running across network) or the https addresses used to reach the web front-end. The list was divided into two groups: one that included IP addresses of technical support; and a second that included all other IP addresses, this second list is referred to as THE INTERNET USERS ACCESSING THE SITE'S ENVIRONMENT.

With no real leads and only a list of users who accessed any host associated with providing the application to users, we called the customer. During the conference call, we asked them if they were comfortable with manually fixing the modification (up to this point no other change had been identified) and re-opening the site to the public. The customer indicated that they saw no immediate problem with the request, but needed to contact other groups, within their organization, to verify that re-opening the site's application to the Internet was appropriate at that time.

So, while waiting for the customer to contact us with their answer, I started analyzing more of the trace data that had been accumulating since just after the intrusion was identified. The traffics were numerous. Several gigabytes of data

were ready to be analyzed. Initially, I looked thru header only data for unusual characteristics in the general flows, but found nothing unusual. Next, I recorded a listing of the IP addresses outside the data center that accessed the IP addresses associated with the application and ran them thru a script to perform reverse DNS look-ups against the customer's primary name server. All of the hosts on the list had reverse entries registered, so again nothing seemed unusual. The listing that now included IP addresses and hosts names was forwarded to the customer. A site representative then verified that all the hosts on the list were running data to/from the data center.

Another conference call was initiated. On the call, the customer indicated that there were no concerns with bringing the site back on-line. During the conference call, within several seconds of bringing site back on-line, the customer notified us that the same table was modified again. It was then agreed that the site needed to be taken back off-line.

- Did a running process coincidentally change the table at the same approximate time that the site was brought back on-line? As far as we were concerned, nothing running before returning the site on-line was inappropriate. So I looked at the running trace files to verify that no running sessions had morphed in the last several seconds and started SELECT INTO, REPLACE or other similar modifications. No morphing was found to have transpired.
- Did a user log into the site about the time the site went back on-line and make the modification? I went thru firewall logs and found one match from the INTERNET USERS ACCESSING THE SITE'S ENVIRONMENT list (see We needed a listing of all IP addresses that had accessed the IP addresses associated with the application within the two weeks preceding the intrusion). Maybe an image of the intruder was forming.

With the one IP address, we went thru logs and other available tools to find out what was accessed by the IP address. We only found that the user went thru the environment using it as was intended. The user logged into a web server, which started a java applet. The user was doing things with the tools made available to him or her on the Internet, nothing at all suspicious. But there had to be more to this, so I queried ARIN, http://www.arin.net/, for information about the intruder's IP address. The IP address was owned by an ISP servicing customers from a state across the country, this seemed odd as the service provided by our customer would typically be much more useful to someone who could within driving distance to the site.

A chance existed that IP address used was being spoofed. So, I performed some O/S finger printing. The device sending the packets was likely an inexpensive router. Next, I verified that traffic, to the intruding IP address, was flowing thru the inexpensive router. So now I decided to see if any ports on the device were open on the Internet. The device allowed telnet traffic (telnet?) and one other port. Was this a huge oversight, a decoy, or even possible? Could it be that a network device had telnet access open to it across an Internet routable IP address? At this point, I opened a telnet session with the device.

The banner indicated a company name and did not even have the unintended use will be prosecuted message. Things were really getting strange now. I contacted the customer and eventually the ISP owning the IP address for follow-up information.

The ISP contacted the end-user, I assume from DHCP log, who was by no means the elusive frontal lobe predator imagined, rather someone who accessed a software application made available to the public by the customer. In the end no foul play was discovered. Just someone who stumbled onto the fact that someone else, who themselves had high privilege on an ERP application, had enabled a default high privilege login with the default password. I personally had used the account numerous times during pre-implementation to generate network flows and prepare expected network characterizations (used repeatedly during the intrusion analysis). The same account was also one of the accounts used to help verify application functionality and performance with the network components implemented via load tests in a test environment.

AFTER

The intrusion occurred even though the network components put in place to service and guard the application functioned as designed, tested and implemented. The intruder gained access to the resources only after the three requirements for access control were passed successfully. The user had an Identity (login). Authentication of the identity by the user was confirmed via a correct password or something he/she knew. And Authorization or permission to access then modify the resource was given to the identity by a third party with the authority to do so granted by the customer.

The intruder did not gain unintended access to the customer's network to get the login or password or to complete any other information system reconnaissance that could be found via logs or other security tools in the several days preceding the intrusion or during the intrusion itself. No host in the customer network was found to have any software, such as Loki or Morpheious or PCAnywhere, frequently used to send reconnaissance information back thru open ports to a possible intruder. In addition, no virus software was found on any hosts in the customer network, although there is no way of verifying, that any such software is/is not running on any of the technical support, located outside of the data center, hosts authorized to access hosts inside the data center. Although it cannot be stated that the intruder never engaged in any unintended information system reconnaissance activities, no proof was found to substantiate the possibility.

Intruder access was provided quickly and securely using the devices intended to provide the application to the Internet. All of the different and separate operating systems and IP abstractions were traversed, firewalls allowed the traffic to pass, valid and authorized identities were confirmed, so that transactions could be queued and completed in a methodical manner as planned, following all rules. In an odd way, the intrusion was successful in part to proper design, planning and execution.

The security design detailed in this document is similar to other separate implementations in the data center. The general methodology has proven itself over time, but has a large over-site that was uncovered during this incident. The over-site being that it only takes one lazy, or under-trained in the security field, high privilege support person to remove not one layer of the onion, but rather renders all layers ineffective. Luckily, the intruder was someone who did not want to corrupt or distribute the sensitive data housed within the application to the Internet, but this was due do to luck rather than plan or execution.

Eradication, for this particular intrusion or intrusions similar to it, the only foreseeable eradication would be for the security group to have access to all changes made inside and outside the application. Granting this type of access or privilege to any person or group is unfathomable, as sooner or later the privilege would be used inappropriately. If power corrupts, then this type of power could corrupt absolutely as one of the first to ever know of any inappropriate use would be the person/group guilty of the act.

The customer completed the incident recovery from the intrusion by returning the table to its state prior to any intrusions by re-keying the changed entries. Excessive amounts of time were spent looking for other changes, but nothing else was discovered as modified. So backups were not needed to for the recovery but just having them available made attempts to discover modifications simpler besides making everyone involved more at ease.

Lessons learned from the intrusion include:

1. Even though, in the back of your mind, you know the eventuality of an intrusion is when rather than if. Don't think that the intrusion will necessarily be a cutting edge exploit or highly skilled adversary. It may likely be a lazy, or unskilled in the field of security, support person circumventing all security for a reason never to be known by the architect/implementer. And the circumvention of security may, as is believed to be the case with this one, occur without collaboration between the intruder and the person circumventing the security.
2. To test for the impossible is not a waste of time. The default login/password combination was not available well before the application initial implementation on the Internet. So to be extra sure one ought to test the impossible with regularity, but there is a lot to test if you try to test the impossible. If you monitor/test the network for availability and performance, too ensure what was intended is available and performing well, AND you test for the impossible THEN there is not enough time in a day. Therefore it is not practical or possible to be extra sure. Contract requirements and due diligence likely demand that you monitor the intended and enforce policy, but the extra part likely falls in the crack and is completed as time permits as opposed to continuously.

Conclusion

After the intrusion I attempted, from a non-trusted IP address, to log into all sites with the default login password that was used in the intrusion. It did not work on any site. We continue to implement networks for customers in the manner that we did before the intrusion, with the belief that sooner or later a real hacker or exploit will cause a compromise. With regards to network security, the main difference between before and after the intrusion is that since the intrusion, especially with Internet-specific enabled applications, we now know that even when everything is working as designed and everybody within the data center is diligent about their craft, applying whatever knowledge and experience they have, all it takes is one person, either intentionally or unintentionally to make it all for not. In fact in the right circumstance the network environment works to help the intruder to complete his or her work quickly and securely.

As mentioned earlier, we did not implement any type of true fix. We did however discover a huge vulnerability that could exist in any implementation of this application. The customer is aware of the vulnerability, but does not want to take any actions to officially monitor for it or guard against it at this point in time. As a data center, we know to check for this vulnerability in any future intrusions for this customer.

On the positive side, the intrusion analysis was beneficial as far as providing an opportunity to actually work within the customer production environment with production data. As mentioned earlier in this paper, we do periodically, with the approval of the customer, make intrusion attempts and perform audits. But the intrusion attempts are via fairly well known applications and documented scriptable vulnerabilities; which the network security is designed/maintained to not be susceptible to. The audits typically validate whether or not procedures are being followed and/or to bring up issues to management. The analysis validated that network security architecture/ implementation performed well. In the end the sensitive data was not disbursed on the Internet, recovery took seconds, and no black hat was found.

Bibliography

1 - http://www.arin.net/