CA AppLogic® application-centric clouds make it easy to secure your applications and data

OVERVIEW

Clouds provide on-demand scalability of highly available applications and shared pools of infrastructure resources, hopefully ensuring secure access to applications, services, and data. Cloud platforms are more uniform than traditional computing centers. From a security perspective, this uniformity enables cloud-wide security mechanisms to protect all data and applications in the cloud. CA AppLogic, an application-centric cloud solution, provides many levels of security that simplify securing both applications and data, increasing the security, resiliency, and robustness of cloud application deployments.

In our tests at Principled Technologies, we found that the CA AppLogic zero-trust security model effectively protected cloud applications from external attacks and that its architecture and application models ensured robust and secure operation.

This paper focuses on the security aspects of CA AppLogic. We tested CA AppLogic security features for version 2.9.9 on four dual-processor Dell™ PowerEdge™ 2950 servers. We configured these servers with

FEBRUARY 2011
A PRINCIPLED TECHNOLOGIES WHITE PAPER
Commissioned by CA Technologies

direct-attached storage drives, and connected them with dual front-end and backend switches and with an Internet accessible input/output security layer.

We examined several aspects of the CA AppLogic cloud security, in particular (1) resiliency and robustness in the presence of cloud controller failures, (2) security in the presence of external attacks to cloud applications, and (3) application internal security as implemented by the cloud interconnect fabric.

INTRODUCTION

Cloud security

The issues in cloud security range from the physical security of the cloud installation and hardware infrastructure, through the architectural security of application and data deployments, to the actual security of the cloud fabric in the presence of external attacks and the mechanisms available to respond to and recover from these attacks. The physical security of the cloud installation is important in any cloud installation, and involves human, physical, and business process considerations. The latter two issues are determined most significantly by a cloud's underlying technologies.

Application and data security starts at the edge of a cloud, with the prevention of security breaches in the firewalls and gateways that provide access to users of cloud resources. The next layer of security involves containing such breaches, by impeding the propagation from an edge component to internal components to compromise them. The third layer of security involves quick recovery to maintain high availability of the cloud to its users. Many technological features enable each layer. For example, scalability offers resilience both in the presence of increased service demands, and when under such attacks as a distributed denial of service (DDoS). In these cases, the ability to scale allows the cloud to react to maintain service levels for its users. Similarly important is the ability to maintain backup instances of applications that are ready to run; whenever the running instance of an application fails, a backup copy can immediately be activated to continue providing services. In a similar way, replicated storage protects data from both physical and logical damage to devices. The uniform, homogenous nature of cloud infrastructure combines to amplify the benefits of each of these technological capabilities.

Another security advantage clouds have over traditional data center deployments comes from the sheer complexity of modern applications. Traditional client-server application deployments that provide shared multi-tenant environments to their users have many potential security issues arising from the sharing of critical application components amongst unrelated users or subscribers, and from the inherent difficulty of

achieving logical separation in multi-tenant situations, which configuration errors or component bugs can compromise. This is not the case for cloud-based virtualized deployments, where separate application instances can be easily created for each client, in completely separate address and data spaces.

CLOUD SECURITY IN CA APPLOGIC

Application and data security in CA AppLogic

CA AppLogic implements a zero-trust security model, where applications are defined as logical flows between virtualized components or appliances, and where the inputs and outputs of each component are defined by its function. Users visually define applications as diagrams in terms of appliance nodes representing gateways, firewalls, load balancers, Web servers, database servers, monitors, etc., and links between these nodes that represent logical connections between appliances. (See Figures 13 and 14 in the What We Tested section for diagrams of typical applications.)

Each appliance is set up as a virtual dedicated server with access restricted to cloud administrators. Links are implemented by the cloud as point-to-point IP virtual network links between appliance VMs. The virtual machines that run an application's appliance are loosely coupled, and each link acts like a virtual firewall on a virtual network connection. This firewalling between every appliance VM part of an application cannot easily be achieved in a real-life data center.

In the zero-trust security model, all links are defined between virtualized appliances. Penetration of an external gateway into an application's deployment cannot extend beyond adjacent appliances. Access to these requires a detailed understanding of the internal application architecture. Furthermore, appliance-appliance connections are protected by cloud controller monitoring. Data packets are only allowed between defined network connections, and packet-by-packet inspection guarantees that rogue packets will be dropped.

What this means is that, in a CA AppLogic application deployment, the traditional hacker attack pattern does not work. In this attack pattern, the hacker penetrates a gateway or edge device in a cloud deployment and searches for configuration files that include names, IP addresses of other application components, scripts, and perhaps even actual passwords. With the addresses, names, and passwords, the attacker hops inside the application into other application components with the intent to reach its core components to compromise the application's operation. In CA AppLogic, all I/O and storage mounts are packaged, and device names are not available. The information required for data flows inside the application is obfuscated inside each virtual appliance, so there is no data available for an attacker to use to deepen an attack. Furthermore, the AppLogic

cloud controller controls all I/O and components as well as tunnels through the network. The moment a packet is launched outside an appliance VM, the cloud controller checks to see whether that packet complies with the corresponding application diagram. If not, the cloud controller drops and ignores the packet.

To increase security further, CA AppLogic provides for the cloud controller to migrate should the server it is running on fail or stop running for any reason. In this case, the cloud controller automatically starts up on any available server, from which it continues providing the needed cloud management and data flow security.

Summary

CA AppLogic has minimized the code necessary to implement application-wide security, thus saving the added risks, effort, testing, and maintenance of other approaches while providing the largest software compatibility footprint. The combination of data replication, application hot-backups, server failure recovery, and other features discussed in depth elsewhere (see the VIRTUALIZED STORAGE IN CA APPLOGIC[1] and CLOUD NETWORKING WITH CA APPLOGIC[2] white papers in this series), combined with the zero-trust security model where the cloud controller decouples the security failure chain and penetration points are only at the edge of the cloud, provides a solid security framework for cloud users. We elaborate on our findings in the next sections.

WHAT WE FOUND

To test the security of CA AppLogic, we set up a test cloud and defined several applications to see how vulnerable they were to outside attacks. We began the evaluation by examining the redundancy of the CA AppLogic cloud controller. Next, to test an application's security, we used external programs to find open network ports and weaknesses in the security. We discuss our findings below.

Initial cloud setup

We began our testing of CA AppLogic by setting up an operational cloud. The most challenging part of the process was ensuring that our physical hardware setup satisfied all constraints imposed by the CA AppLogic cloud controller, which involved our studying in depth the available documentation and performing several experiments. Once we achieved this, deploying the cloud was simple and involved only a minimal amount of work. We expect that, for most users, a successful first CA AppLogic cloud deployment will make

[1] http://principledtechnologies.com/clients/reports/CA/AppLogic_VSAN.pdf
[2] http://principledtechnologies.com/clients/reports/CA/AppLogic_networking.pdf

subsequent deployments much simpler as users will better understand the physical hardware requirements of their clouds.

We decided to build our cloud using commodity Linux servers. We selected one of these as our deployment server, called the *Aldo Distribution Server* in the CA AppLogic documentation. We downloaded the CA AppLogic software onto it and set up the software. Next, we defined a cloud configuration file, which included a number of required attributes at the cloud boundary, including the name of the cloud, the IP address range for external IPs, a list of IP addresses of all servers to use for deployment of the cloud, a Secure Shell (SSH) certificate path, and a user name associated with the certificate. Defining the configuration file was straightforward. The SSH certificate needed to be set up from the outset because it serves the role of a license key—it is authenticated by the CA AppLogic servers, verifies that an AppLogic cloud deployment is valid, and is required to download updates from the CA AppLogic Web site.

Before you can install CA AppLogic on a cloud server, it must be running a Linux distribution. CA has verified that AppLogic works with Red Hat®, CentOS, and Fedora™ distributions. We had CA AppLogic install 32- and 64-bit versions of CentOS initially, and later we had it add a Red Hat Enterprise Linux® 5.5 server to our cloud.

Once we defined the configuration file and our deployment and cloud servers were in place, we ran a CA AppLogic command to create the cloud from the configuration file. This installed a bare-minimum Linux configuration on each of the cloud servers and created two partitions in each cloud server with the CA AppLogic cloud software, one active and one backup. This feature allows you to roll back to a previous version of CA AppLogic if a software upgrade does not work as expected.

At this point, CA AppLogic built a VSAN accessible from all directly attached storage in the cloud servers. During this process, CA AppLogic selected one of the cloud servers as the primary server for the cloud controller and one or more secondary servers as backups should the primary server fail. Once CA AppLogic had finished deploying the cloud, we verified that the cloud was operational and that we could log in and start defining our test application. Figure 1 shows the cloud test environment we built.

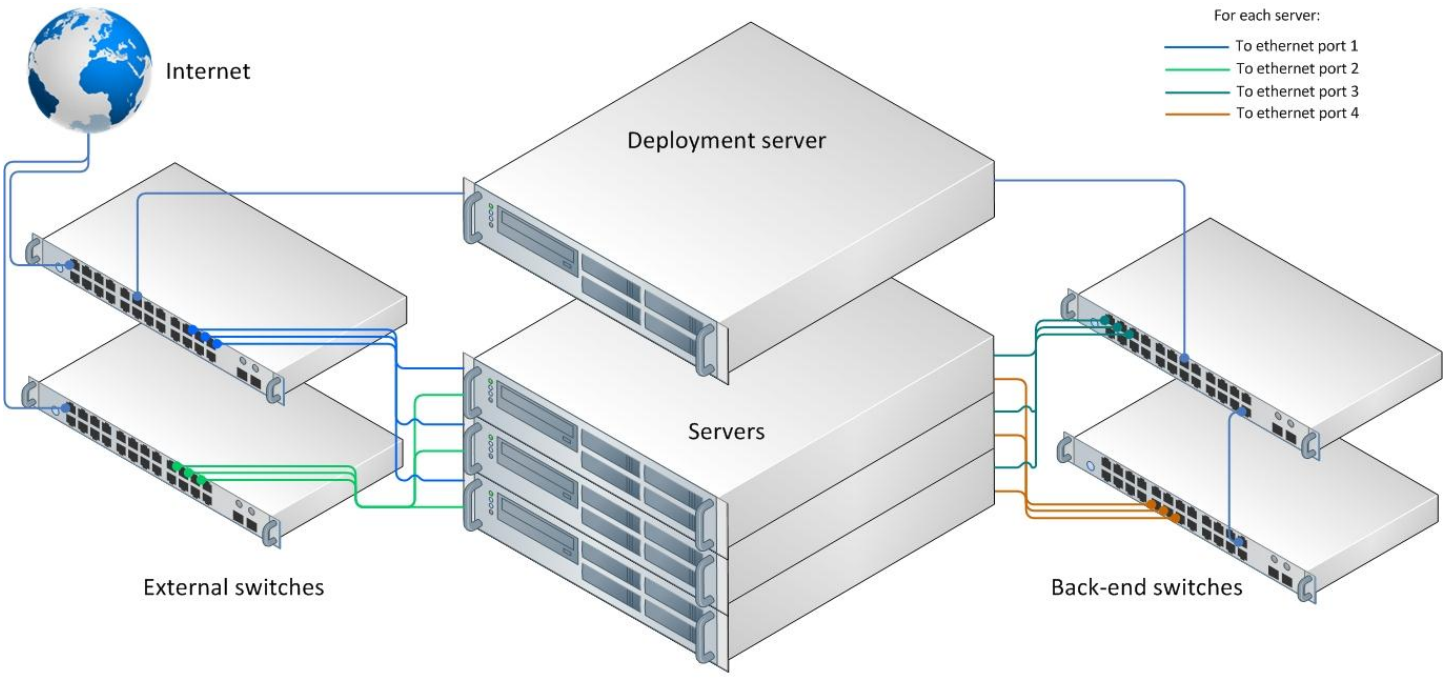

Figure 1: Our test environment.

Cloud controller redundancy

The CA AppLogic cloud controller holds the configuration information for the applications and storage volumes. To examine the effects on the cloud of a complete system failure of the cloud controller, we induced such a failure by pulling the power cord from the server.

Prior to our pulling the server's power cord, we used PuTTY, a freeware Secure Shell (SSH) client, to log into the cloud controller. We then used the `3t` command to start a grid shell inside the SSH session. Once the session started, we used the `srv list` command to display a list of all servers in our cloud; this list indicated which server was filling the primary role. Figure 2 displays the output of this command prior to our introducing the failure.

```
(root)pt-test> srv list
Name     State          CPU            Mem(MB)      BW (Mbps)     Role
                    Alloc  Free     Alloc  Free    Alloc  Free
srv1     up          0.00   8.00         0  15619       0  2000    secondary
srv2     up          0.00   7.90         0  14851       0  2000    primary
srv3     up          0.00   8.00         0  15619       0  2000    secondary
```

Figure 2: Output of the `srv list` command prior to our introducing the failure.

As Figure 2 shows, srv2 was the primary cloud controller. The other two, srv1 and srv3, acted as secondary servers in the event of a failure on the primary server. Note: CA AppLogic defines only two servers as secondary, so a larger cloud configuration would not show all servers in a secondary role.

Next, we introduced a catastrophic failure by pulling the power cord from srv2. After the server turned off from the power failure, a message appeared on the CA AppLogic dashboard reporting a hardware fault (see Figure 3).

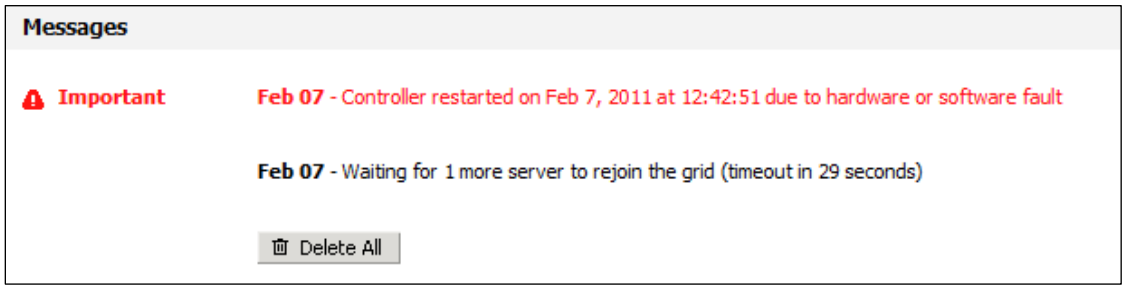

Figure 3: Hardware fault message on the CA AppLogic dashboard.

After the fault message appeared, we used the same method as before to establish an SSH session with the cloud controller and show a server list. The list showed srv2 status as unknown and changed its role to secondary. Srv1 was moved to primary role. Figure 4 displays the output of the `srv list` command after the failure.

```
(root)pt-test> srv list
Name     State          CPU            Mem(MB)      BW (Mbps)     Role
                    Alloc  Free     Alloc  Free    Alloc  Free
srv1     up          0.00   7.90         0  14851       0  2000   primary
srv2     unknown     0.00   0.00         0      0       0     0   secondary
srv3     up          0.00   8.00         0  15619       0  2000   secondary
```

Figure 4: Output of the `srv list` command after the failure.

Our testing demonstrates that the cloud worked as expected and, in the event of a failure on the primary cloud controller, switched to one of the two secondary servers.

Application external security

Next, we looked at the security of CA AppLogic cloud applications to external network traffic. We used Nmap, an open-source network security scanner, to check for any weaknesses. We configured a redundant Web server application and a redundant database application to check for vulnerabilities. The How We Tested section of this report provides configuration information for each application.

We began by scanning the redundant Web server application. We used the `nmap -v -p 1-65535 172.16.84.102` command for the scan. Figure 5 shows the Nmap report from the scan. As it shows, port 80 was the only open port. Port 80 is used for all Web traffic from the server, so it must be open. Figure 5 shows the Nmap results; Appendix B provides the complete Nmap output, including scan times and total packets sent during the test.

```
Nmap scan report for 172.16.84.102
Host is up (0.00s latency).
Not shown: 65534 filtered ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: F2:01:01:00:2E:07 (Unknown)
```

Figure 5: Nmap report for the redundant Web server application scan.

Next, we scanned the redundant database server application. We used the same command, substituting the IP address of the database application. Figure 6 shows the Nmap report from the scan. As it shows, port 3306 was the only open port. Port 3306 is used for database traffic, so it must be open. Figure 6 shows the Nmap results; Appendix C provides the complete Nmap output.

```
Nmap scan report for 172.16.84.112
Host is up (0.00s latency).
Not shown: 65534 filtered ports
PORT     STATE SERVICE
3306/tcp open  mysql
MAC Address: F2:01:01:00:40:29 (Unknown)
```

Figure 6: Nmap report for the redundant database server application scan.

Our testing demonstrates that the ports specified by the application configuration were the only open ports. Thus, the application left no additional ports open for potential unwanted traffic or security breaches.

Application internal security

We evaluated the security of CA AppLogic cloud applications on the internal network by looking at routing tables and performing ping tests to ensure communication. We configured a simple application with a gateway, Web server, and database server to check for vulnerabilities. Figure 7 shows the simple application; the How We Tested section of this report provides additional configuration information.

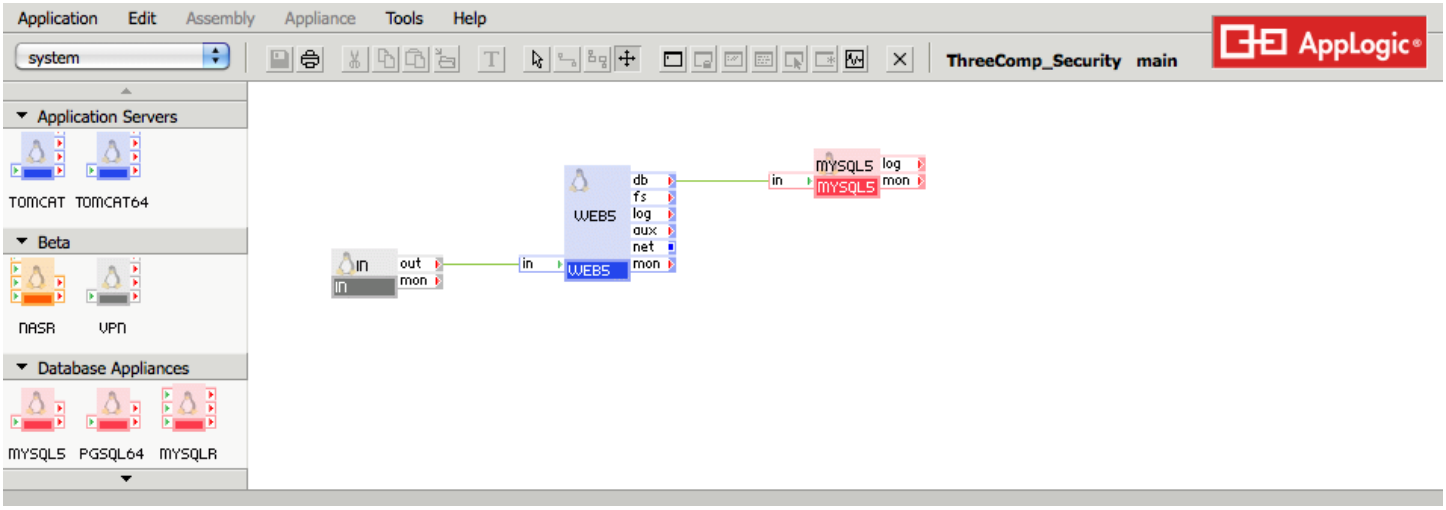

Figure 7: The simple Web server and database server application we used for our evaluation.

Figure 8 shows the IP addresses and routing table for the *in* appliance used as a gateway. The *in* appliance has four network connections: eth0 is the external connection, eth1 is the out port used to connect to another appliance, eth2 connects to the system monitor, and eth3 connects to the cloud controller. The routing table shows the destination IP addresses.

```
IN ipaddr
eth0: 172.16.84.106/24
eth1: 10.8.2.1/32
eth2:
eth3: 10.8.2.5/13

IN route
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.2     *               255.255.255.255 UH    0      0        0 eth3
10.8.2.2        *               255.255.255.255 UH    0      0        0 eth1
172.16.84.0     *               255.255.255.0   U     0      0        0 eth0
169.254.0.0     *               255.255.0.0     U     0      0        0 eth0
10.8.0.0        *               255.248.0.0     U     0      0        0 eth3
default         172.16.84.1     0.0.0.0         UG    0      0        0 eth0
```

Figure 8: IP address and routing table for in gateway appliance.

Figure 9 shows the IP addresses and routing table for the Web server appliance. The Web server appliance has eight total network connections, but we only configured eth0 and eth1 for testing. The third IP address shown, eth7, is used to connect to the cloud controller.

```
WEB ip addr
eth0: 10.8.2.2/32
eth1: 10.8.2.3/32
eth7: 10.8.2.7/13

WEB route
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.8.2.4        *               255.255.255.255 UH    0      0        0 eth1
192.168.1.2     *               255.255.255.255 UH    0      0        0 eth7
10.8.2.1        *               255.255.255.255 UH    0      0        0 eth0
10.8.0.0        *               255.248.0.0     U     0      0        0 eth7
```

Figure 9: IP address and routing table for Web server appliance.

Figure 10 shows the IP addresses and routing table for the database server appliance. The database server appliance has four network connections: eth0 is the connection from the Web server, eth1 is the port used for logging, eth2 connects to the system monitor, and eth3 connects to the cloud controller. We did not configure eth1 or eth2 for this testing.

```
MYSQL ip addr
eth0: 10.8.2.4/32
eth1:
eth2:
eth3: 10.8.2.6/13

MYSQL route
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.2     *               255.255.255.255 UH    0      0        0 eth3
10.8.2.3        *               255.255.255.255 UH    0      0        0 eth0
10.8.0.0        *               255.248.0.0     U     0      0        0 eth3
```

Figure 10: IP address and routing table for database server appliance.

We performed a ping test from the different appliances to see if their network paths coincided with routing tables. We connected to the cloud controller with a SSH connection using PuTTY. Once connected to the cloud controller, we used the `3t` command to start a grid shell inside the SSH session. Once the session started, we used the `SSH my_app:main.appliance` command, where "my_app" is the name of the

application and “main.appliance” is the name of the appliance you are connecting to. Figure 11 shows the results from the ping test.

```
      ping WEB to IN
[ThreeComp_Security:main.WEB5 ~]# ping -c 4 10.8.2.1
PING 10.8.2.1 (10.8.2.1) 56(84) bytes of data.
64 bytes from 10.8.2.1: icmp_seq=1 ttl=64 time=0.109 ms
64 bytes from 10.8.2.1: icmp_seq=2 ttl=64 time=0.063 ms
64 bytes from 10.8.2.1: icmp_seq=3 ttl=64 time=0.071 ms
64 bytes from 10.8.2.1: icmp_seq=4 ttl=64 time=0.061 ms
--- 10.8.2.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms`
rtt min/avg/max/mdev = 0.061/0.076/0.109/0.019 ms

ping WEB to MYSQL
[ThreeComp_Security:main.WEB5 ~]# ping -c 4 10.8.2.4
PING 10.8.2.4 (10.8.2.4) 56(84) bytes of data.
64 bytes from 10.8.2.4: icmp_seq=1 ttl=64 time=7.18 ms
64 bytes from 10.8.2.4: icmp_seq=2 ttl=64 time=0.059 ms
64 bytes from 10.8.2.4: icmp_seq=3 ttl=64 time=0.064 ms
64 bytes from 10.8.2.4: icmp_seq=4 ttl=64 time=0.063 ms
--- 10.8.2.4 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3007ms
rtt min/avg/max/mdev = 0.059/1.843/7.188/3.085 ms

ping MYSQL to IN
[ThreeComp_Security:main.MYSQL5 ~]# ping -c 4 10.8.2.1
PING 10.8.2.1 (10.8.2.1) 56(84) bytes of data.
--- 10.8.2.1 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3000ms
```

Figure 11: Ping test results for the three appliances.

As the results show, the ping test lines up with the routing table guidelines. There was communication between the *in* gateway and the Web server appliances, as well as communication between the Web server and database appliances, but not between the database and in gateway appliances. This shows that

communication goes through only the outlined paths held in the cloud controller and displayed in the routing tables.

HOW WE TESTED

One advantage of using CA AppLogic is that the user does not need identical server configurations for his or her cloud infrastructure. For testing, we used four Dell PowerEdge 2950 servers with different hardware configurations. Figure 1 in the What We Found section illustrates our test environment. Appendix A provides detailed configuration information for the test servers.

Configuring the redundant Web server application

Figure 12 shows the redundant Web server application we used for application external security testing. We dragged all components from the left panes to the application editor main window. We connected all components as we outlined.

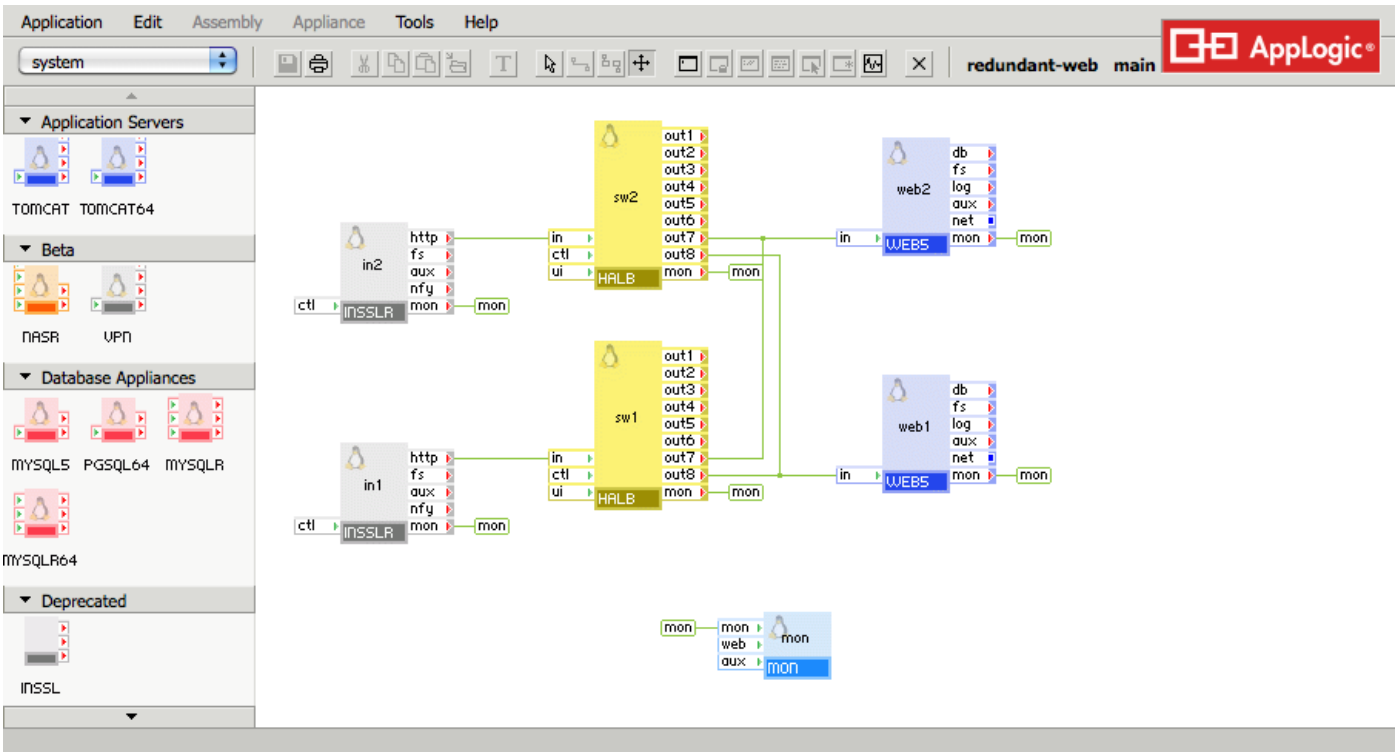

Figure 12: The redundant Web server application we used for our evaluation.

Configuring the gateways

We made the following configuration options to the *in1* and *in2* gateways:

- ip_addr: `172.16.84.102`
- netmask: `255.255.255.0`
- gateway: `172.16.84.1`
- l7_accept: `http`
- fover_mode: `symmetric`
- fover_local_ip: `192.168.100.1 (in1); 192.168.100.2 (in2)`
- fover_remote_ip: `192.168.100.2 (in1); 192.168.100.1 (in2)`
- fover_netmask: `255.255.255.0`

Configuring the redundant database server application

Figure 13 shows the redundant database server application we used for application external security testing.

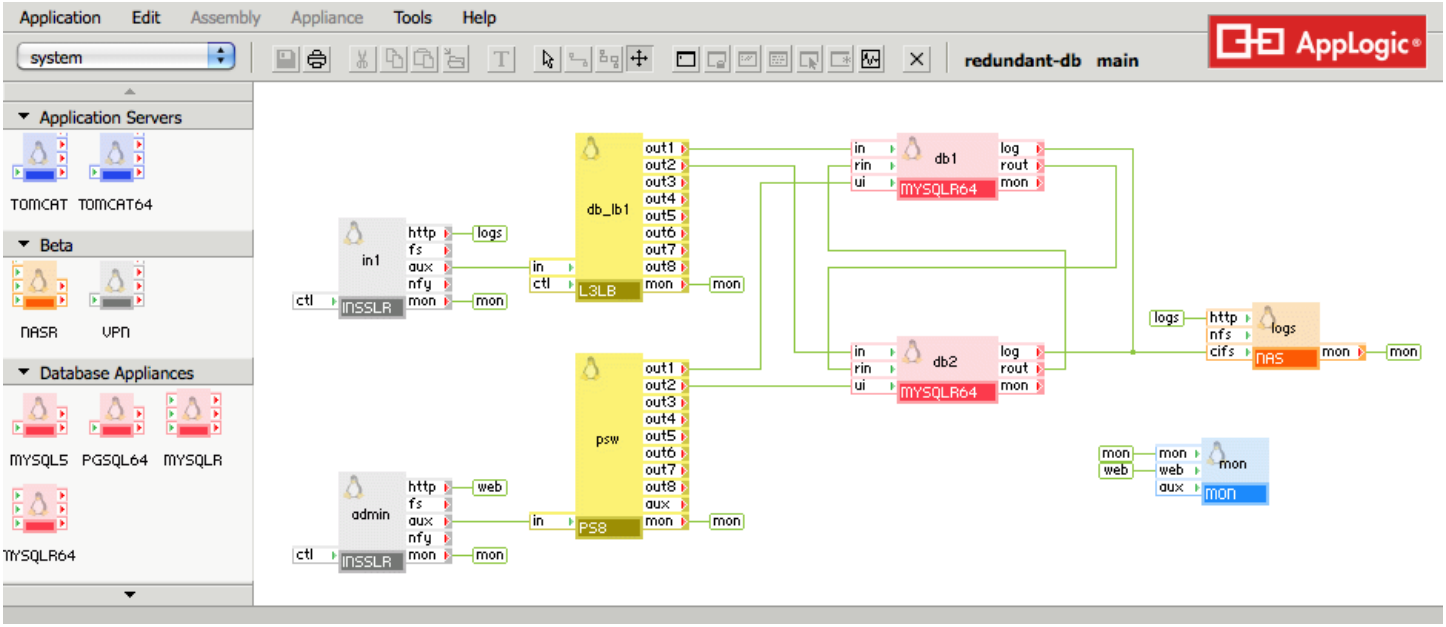

Figure 13: The redundant database server application we used for our evaluation.

Configuring the gateways

We added two gateways to our application, one for test traffic and one for administrative functions. We made the following configuration changes to the gateways:

- ip_addr: `172.16.84.112 (in1); 172.16.84.113 (admin)`
- netmask: `255.255.255.0`
- gateway: `172.16.84.1`
- l7_accept: `http`

- l3_accept_proto: `tcp`
- l3_accept_port: `3306`

Configuring the load balancer

We used a L3LB – TCP/UDP load balancer application to evenly distribute the workload across the two database servers. We made the following configuration change to the load balancer:

- ports: `3306`

Configuring the volume

We used the Manage Volume Configuration window to set up the virtual storage for testing. We created six different ext3 data volumes for the application. The volume names and sizes are as follows:

- Db1_binlogs: `2GB`
- Db1_data: `8GB`
- Db2_binlogs: `2GB`
- Db2_data: `8GB`
- Logs_data: `100MB`
- Mon_data: `100MB`

Configuring the database

We added two database servers to the application. We made the following configuration changes on both database servers:

- error_log_filename: `db1.log(Server 1); db2.log(Server 2)`
- error_log_level: `warn`
- timezone: `EST`
- server_id: `1 (Server 1); 2 (Server 2)`
- rpl_mode: `master_and_slave`
- web_pwd: `password`

Configuring the NAS appliance

We added a NAS appliance for log data. We made the following configuration changes:

- http_dir_enabled: `yes`
- timezone: `EST`

Configuring the port switch

We added a PS8 cascadable port switch to the application for administrative purposes. We made the following configuration changes:

- out1_protocol: `tcp`
- out1_in_port: `8081`
- out1_out_port: `http`
- out2_protocol: `tcp`
- out2_in_port: `8082`
- out2_out_port: `http`

Configuring the monitor

We added an application-monitoring appliance to the application so we could monitor system activity during testing. We made the following configuration changes:

- user: `admin`
- password: `password`
- alarm_view: `db_alarm`

Configuring the simple Web and database server application

Figure 14 shows the simple Web and database server application we used for application internal security testing.

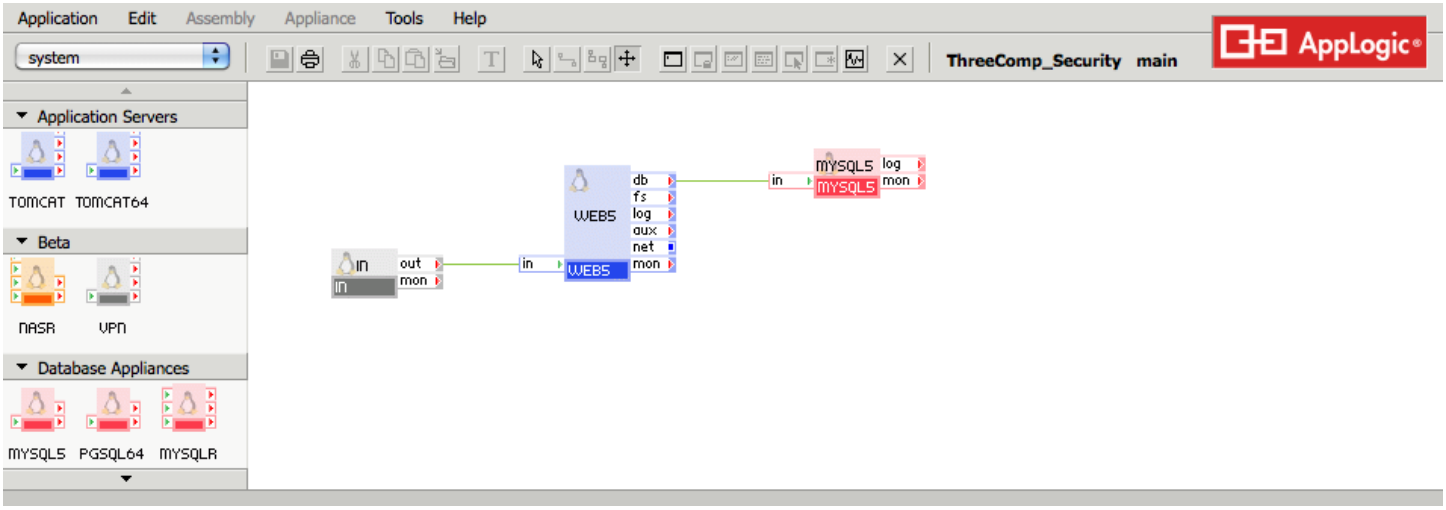

Figure 14: The simple Web and database server application we used for our evaluation.

Configuring the gateway

We added two gateways to our application, one for test traffic and one for administrative functions. We made the following configuration changes to the gateways:

- ip_addr: `172.16.84.106`
- netmask: `255.255.255.0`
- gateway: `172.16.84.1`

APPENDIX A – SERVER CONFIGURATION INFORMATION

Figure 15 provides detailed configuration information about the test servers.

System	Dell PowerEdge 2950 server 1	Dell PowerEdge 2950 server 2	Dell PowerEdge 2950 server 3	Dell PowerEdge 2950 server 4
Power supplies				
Total number	1	1	1	1
Vendor and model number	Dell 7001072-Y000	Dell 7001072-Y000	Dell 7001072-Y000	Dell 7001072-Y000
Wattage of each (W)	750	750	750	750
Cooling fans				
Total number	4	4	4	4
Vendor and model number	NMB-MAT 2415KL-04W-B96	NMB-MAT 2415KL-04W-B96	NMB-MAT 2415KL-04W-B96	NMB-MAT 2415KL-04W-B96
Dimensions (h x w) of each	2” x 2”	2” x 2”	2” x 2”	2” x 2”
Volts (V)	12	12	12	12
Amps (A)	2.10	2.10	2.10	2.10
General				
Number of processor packages	2	2	2	2
Number of cores per processor	4	4	4	4
Number of hardware threads per core	1	1	1	1
CPU				
Vendor	Intel®	Intel	Intel	Intel
Name	Xeon®	Xeon	Xeon	Xeon
Model number	E5440	E5405	E5405	E5405
Socket type	LGA771	LGA771	LGA771	LGA771
Core frequency (GHz)	2.83	2.00	2.00	2.00
Bus frequency (MHz)	1,333	1,333	1,333	1,333
L1 cache (KB)	128	128	128	128
L2 cache (MB)	12	12	12	12
Platform				
Vendor and model number	Dell PowerEdge 2950	Dell PowerEdge 2950	Dell PowerEdge 2950	Dell PowerEdge 2950
Motherboard model number	0J250G	0M332H	0M332H	0M332H
Motherboard chipset	Intel 5000X	Intel 5000X	Intel 5000X	Intel 5000X
BIOS name and version	Dell 2.6.1	Dell 2.6.1	Dell 2.6.1	Dell 2.6.1
BIOS settings	Default	Default	Default	Default

System	Dell PowerEdge 2950 server 1	Dell PowerEdge 2950 server 2	Dell PowerEdge 2950 server 3	Dell PowerEdge 2950 server 4
Memory module(s)				
Total RAM in system (GB)	32	16	16	16
Vendor and model number	Hynix HYMP151F72CP4 N3-Y5	Samsung M395T5750EZ4-CE66	Samsung M395T5750EZ4-CE66	Samsung M395T5750EZ4-CE66
Type	PC2-5300F	PC2-5300F	PC2-5300F	PC2-5300F
Speed (MHz)	667	667	667	667
Speed running in the system (MHz)	667	667	667	667
Timing/latency (tCL-tRCD-tRP-tRASmin)	5-5-5-15	5-5-5-15	5-5-5-15	5-5-5-15
Size (GB)	4	2	2	2
Number of RAM module(s)	8	8	8	8
Chip organization	Double-sided	Double-sided	Double-sided	Double-sided
Rank	Dual	Dual	Dual	Dual
Hard disk				
Vendor and model number	Western Digital WD1602ABKS	Western Digital WD1602ABKS	Seagate ST973402SS	Western Digital WD1602ABKS
Number of disks in system	2	2	3	3
Size (GB)	160	160	73	160
Buffer size (MB)	16	16	16	16
RPM	7,200	7,200	10,000	7,200
Type	SATA	SATA	SAS	SATA
Disk controller	Dell SAS 6/IR	Dell SAS 6/IR	Dell PERC 6/I	Dell SAS 6/IR
Operating system				
Name	CentOS Release 5.5	CentOS Release 5.3	CentOS Release 5.3	CentOS Release 5.3
Kernel release	2.6.18-194.el5	2.6.18.8-xen0	2.6.18.8-xen0	2.6.18.8-xen0
Kernel version	SMP Fri Apr 2 14:58:14	#1 Thu May 13 11:28:43	#1 Thu May 13 11:28:43	#1 Thu May 13 11:28:43
File system	ext3	ext3	ext3	ext3
Language	English	English	English	English
Ethernet				
NIC Type 1				
Vendor and model number	Broadcom® 5708 NetXtreme® II 5708 Ethernet Adapter	Broadcom 5708 NetXtreme II 5708 Ethernet Adapter	Broadcom 5708 NetXtreme II 5708 Ethernet Adapter	Broadcom 5708 NetXtreme II 5708 Ethernet Adapter
Type	Onboard	Onboard	Onboard	Onboard

System	Dell PowerEdge 2950 server 1	Dell PowerEdge 2950 server 2	Dell PowerEdge 2950 server 3	Dell PowerEdge 2950 server 4
NIC Type 2				
Vendor and model number	N/A	Intel 82571EB Quad Port Low Profile Adapter	Intel 82571EB Quad Port Low Profile Adapter	Broadcom 5709 NetXtreme II Ethernet Adapter
Type	N/A	PCI-E	PCI-E	PCI-E
Optical drive(s)				
Vendor and model number	TEAC CD-ROM CD-224E-N	TEAC CD-ROM CD-224E-N	TEAC CD-ROM CD-224E-N	TEAC CD-ROM CD-224E-N
USB ports				
Number	4	4	4	4
Type	2.0	2.0	2.0	2.0

Figure 15: Configuration details for the test servers.

APPENDIX B – WEB SERVER NMAP OUTPUT

Figure 16 shows the outputs from the Nmap scans on the redundant Web server application.

```
C:\Program Files\Nmap>nmap -v -p 1-65535 172.16.84.102
Starting Nmap 5.50 ( http://nmap.org ) at 2011-02-07 08:05 Eastern Standard Time
Initiating ARP Ping Scan at 08:05
Scanning 172.16.84.102 [1 port]
Completed ARP Ping Scan at 08:05, 0.05s elapsed (1 total hosts)
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
Try using --system-dns or specify valid servers with --dns-servers
Initiating SYN Stealth Scan at 08:05
Scanning 172.16.84.102 [65535 ports]
Discovered open port 80/tcp on 172.16.84.102
SYN Stealth Scan Timing: About 18.62% done; ETC: 08:08 (0:02:16 remaining)
SYN Stealth Scan Timing: About 44.57% done; ETC: 08:07 (0:01:16 remaining)
Completed SYN Stealth Scan at 08:07, 111.56s elapsed (65535 total ports)
Nmap scan report for 172.16.84.102
Host is up (0.00s latency).
Not shown: 65534 filtered ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: F2:01:01:00:2E:07 (Unknown)
Read data files from: C:\Program Files\Nmap
Nmap done: 1 IP address (1 host up) scanned in 111.73 seconds
           Raw packets sent: 131154 (5.771MB) | Rcvd: 86 (3.768KB)
```

Figure 16: Nmap report for the redundant Web server application scan.

APPENDIX C – DATABASE SERVER NMAP OUTPUT

Figure 17 shows the outputs from the Nmap scans on the redundant database server application.

```
C:\Program Files\Nmap>nmap -v -p 1-65535 172.16.84.112
Starting Nmap 5.50 ( http://nmap.org ) at 2011-02-08 14:18 Eastern Standard Time
Initiating ARP Ping Scan at 14:18
Scanning 172.16.84.112 [1 port]
Completed ARP Ping Scan at 14:18, 0.05s elapsed (1 total hosts)
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
Try using --system-dns or specify valid servers with --dns-servers
Initiating SYN Stealth Scan at 14:18
Scanning 172.16.84.112 [65535 ports]
Discovered open port 3306/tcp on 172.16.84.112
SYN Stealth Scan Timing: About 18.64% done; ETC: 14:21 (0:02:15 remaining)
SYN Stealth Scan Timing: About 44.48% done; ETC: 14:20 (0:01:16 remaining)
Completed SYN Stealth Scan at 14:20, 111.44s elapsed (65535 total ports)
Nmap scan report for 172.16.84.112
Host is up (0.00s latency).
Not shown: 65534 filtered ports
PORT     STATE SERVICE
3306/tcp open  mysql
MAC Address: F2:01:01:00:40:29 (Unknown)
Read data files from: C:\Program Files\Nmap
Nmap done: 1 IP address (1 host up) scanned in 111.61 seconds
           Raw packets sent: 131155 (5.771MB) | Rcvd: 87 (3.812KB)
```

Figure 17: Nmap report for the redundant database server application scan.

ABOUT PRINCIPLED TECHNOLOGIES

Principled Technologies, Inc.
1007 Slater Road, Suite 300
Durham, NC, 27703
www.principledtechnologies.com

We provide industry-leading technology assessment and fact-based marketing services. We bring to every assignment extensive experience with and expertise in all aspects of technology testing and analysis, from researching new technologies, to developing new methodologies, to testing with existing and new tools.

When the assessment is complete, we know how to present the results to a broad range of target audiences. We provide our clients with the materials they need, from market-focused data to use in their own collateral to custom sales aids, such as test reports, performance assessments, and white papers. Every document reflects the results of our trusted independent analysis.

We provide customized services that focus on our clients' individual requirements. Whether the technology involves hardware, software, Web sites, or services, we offer the experience, expertise, and tools to help our clients assess how it will fare against its competition, its performance, its market readiness, and its quality and reliability.

Our founders, Mark L. Van Name and Bill Catchings, have worked together in technology assessment for over 20 years. As journalists, they published over a thousand articles on a wide array of technology subjects. They created and led the Ziff-Davis Benchmark Operation, which developed such industry-standard benchmarks as Ziff Davis Media's Winstone and WebBench. They founded and led eTesting Labs, and after the acquisition of that company by Lionbridge Technologies were the head and CTO of VeriTest.