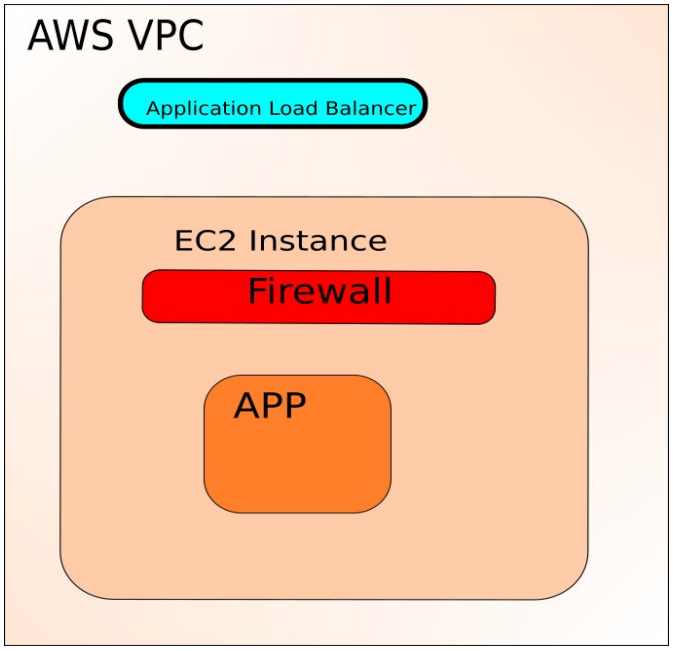
PROPOSAL FOR AMAZIN INC PCI-COMPLIANT AWS INFRASTRUCTURE:

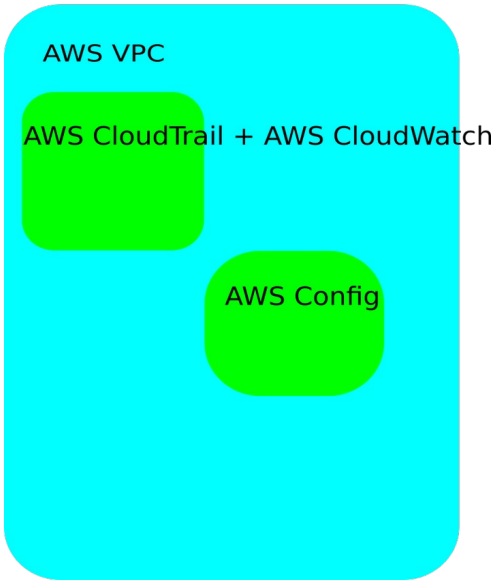
This architecture is composed of the following:

1. **Main production web application Instances(EC2) VPC 1:**

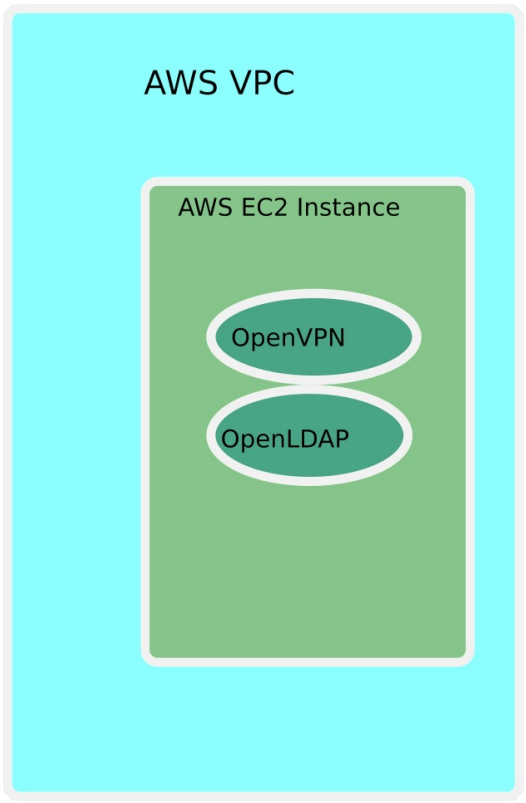


2. **Centralised Logging architecture VPC 2:**

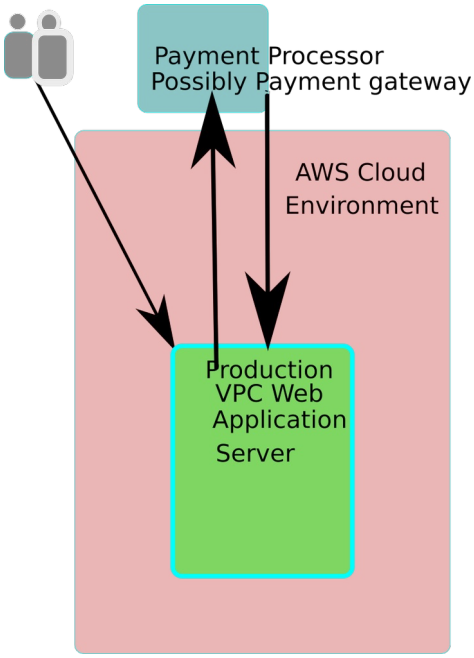
Monitoring and Metrics:



3. Management VPC VPC3:
This refers to the EC2 OpenVPN instance server which has OpenVPN running to allow secure access to the Amazon VPC that has the production servers and client applications for administrative tasks. **Alternatively** this could be done via a secured AWS ***bastion*** host that facilitates command-line Secure Shell access to the EC2 instances for troubleshooting and systems administrations activities.



WHOLE VIEW OF ARCHITECTURE:



It is important to ***reduce the scope of PCI compliance*** by not storing cardholder data as often having these details can lead to greater overhead and is a liability.
This liability extends beyond the risk of losing a card number to the internet and the associated public relations damage.
Properly protecting cardholder data slows down business operations and if it doesn't earn the business money is not worth it.
The costs of having a payment aggregator or a payment gateway (such as Stripe) or a merchant account handling the process of storing this cardholder data and tokenizing the PAN(Personal Account Number) as well as doing things like fraud analysis and detection is not that high as to make managers make the decision to do all this processing on their servers and store this very sensitive data on their databases.

AWS SERVICES USED (BRIEF OVERVIEW):

i. ***AWS CloudTrail:*** AWS CloudTrail records AWS API calls and delivers log files that include caller identity management, time, source IP address, request parameters, and response elements. These details enable security analysis, resource change tracking and compliance auditing.
ii. ***AWS CloudWatch:*** This is a monitoring service for AWS cloud resources and the applications running on AWS. Amazon CloudWatch collects and tracks metrics, collects and monitors log files, sets alarms, and automatically reacts to changes to your AWS cloud landscape/resources.
iii. ***AWS Config:*** This provides resource inventory, configuration history and configuration change notifications to enable security and governance.
iv. ***Amazon EC2 instances:*** These enable the launch of virtual machine instances from a variety of various operating systems. They can be either Amazon EBS-backed EC2 instances or Amazon Instance Store-backed EC2 instances, each with different features.
v. ***AWS Elastic Load Balancing:*** This automatically distributes traffic across multiple EC2 instances to help achieve better fault tolerance and high availability.
vi. ***Amazon VPC:*** The Amazon Virtual Private Cloud lets its users provision a private, logical isolated section of the AWS Cloud where you can launch AWS services and other resources in a virtual network that you define. Hence it enables complete control over your virtual networking environment, including the selection of your own IP address range, creation of subnets, and configuration of route tables and network gateways.
vii. ***(Optional) Amazon CloudFront :*** Given that Amazin is deploying a web application, it can be very useful depending on pricing requirements and location of customers, to have CloudFront set up to cache web content across distributed areas that are located far from the originating servers and also to offload the origin servers.

GENERAL OVERVIEW OF ARCHITECTURE:

- ***Customer checkout:*** The web application will have a checkout or pay button that takes the merchant's (Amazin) customer to the checkout form to fill in data such as his/her personal account number, name on the card, expiry date of card and possibly the CID/CAV2(found at the back of the card or at the front(for American Express/AMEX)).
- ***Segmented secure cloud infrastructure:*** In the cloud, this web app lies in a segmented and secured cloud environment. This segmentation is achieved through the use of Amazon AWS VPC or Virtual Public Clouds which have associated private/public subnets with them. There could be various web apps running on various Amazon EBS-backed EC2 instances each connected to an Application or network load balancer. The application load balancer is preferred in this case. The EC2 instances have autoscaling enabled to grow with increased customer traffic.

- ***Firewall(IPtables, pfSense or AWS WAF) (PCI DSS Requirement 1):*** For security, these EC2 instances need to be configured with a firewall that disables all connections apart from that with the load balancer and they're assigned private IP addresses.
  The ALB (Application Load Balancer) will accept HTTPS connections from customers and hence will have a public IP unlike the EC2 instances running the application code.
  AWS offers AWS WAF or Web Application Firewall but there are also other Open source alternatives such as Iptables which uses netfilter, as well as pfSense.
- ***Reducing PCI Compliance Scope by using a payment gateway/payment aggregator/merchant account:*** It is important that none of this cardholder data is stored in Amazin's servers or in any database and as such only verification of payment is needed from the payment gateway. After the payment gateway processes the transaction(e.g. shoes bought on the ecommerce platform), it sends all the information for approval to the issuing bank via the acquiring bank and relevant card schemes (VISA, Mastercard, AMEX, etc). After approval, once the payment has been confirmed by the card schemes, the payment gateway sends the approved transaction back to the merchant's website (Amazin in this case) and then the merchant informs the customer that the purchase has been successfully completed. If approval fails, this information is also shown on the website to the customer.
- ***OpenVPN:*** Administrative tasks such as management of the instances and configuration editing, patch management are done through an OpenVPN server running on an EC2 Instances that is highly secured.
- ***Logging and monitoring with AWS CloudTrail, CloudWatch and Config(PCI DSS Requirement 10):*** Logging, monitoring, alerts and metrics are done through AWS CloudTrail, CloudWatch and AWS Config.
- **Access Controls(PCI DSS Requirement 7):**For access controls or ACLs, the first thing that is needed is to protect the AWS account. This is done using the Identity and Access management (IAM) and multifactor authentication.
  The root AWS account should never be used, and should be protected by multifactor authentication as well, in addition to strong passwords.
  OpenLDAP provides a good choice as a open source Access control software and is the choice I made.
- ***Anti-virus Software:*** The OpenSource ClamAV antivirus is a good choice for an antivirus.
- ***Intrusion Detection Systems(IDS):*** For intrusion detection, there are numerous open-source solutions such as OSSEC. OSSEC is a host-based intrusion detection system (IDS) with file system integrity monitoring and is backed by Trend Micro.
  Snort is a good choice for a network intrusion detection system (IDS). Snort will use an Amazon RDS database.
- ***Proper Key Management:*** To protect keys and ensure key rotation, AWS KMS is a very good choice and is affordable. It also supports powerful audit trail through AWS CloudTrail.
- (Optional) It is possible to have AWS CloudFront to perform tasks like caching web content and reducing the origin's servers workload. AWS would then connect via HTTP to an S3 bucket or an Amazon EC2 web server that has the web content.

SECURING THE EC2 INSTANCE LINUX VMS:

- PCI DSS 1.1 requires that a merchant establish firewall and router configuration that includes a formal process for approving and testing all network connections and changes to the firewall and router configuration changes, current network diagram that identifies all connections between the cardholder data environment and other networks, description of groups, roles and responsibilities for management of network components, among other specifications.

Iptables is often available in most Linux systems and is the most ubiquitous Linux firewall. Another great firewall is pfSense.
PCI 1.1.6b states that there should be documentation for all security features implemented for those protocols considered insecure. Hence we need to identify insecure services and disable these services. Where possible, insecure services should be replaced with a more secure alternative. For example, all Linux distributions use now SSH by default for remote administration. A protocol like telnet should therefore no longer be used. The “r” services like rexec, rlogin, rcp, and rsh, are insecure as well.

- Unencrypted protocols include FTP, HTTP, IMAP, POP3, SNMP v1/v2 and Telnet which should all be replaced by their secure alternatives, namely FTPS/SFTP/SCP, HTTPS, IMAPS, POP3S, SNMP v3 and SSH or Mosh respectively.
  To determine which protocols are running in an instance, netstat or ss utilities can be used.
  For netstat(On Ubuntu):
  ```
  sudo apt install net-tools
  netstat -nlp
  ```
  On RHEL, Fedora:
  dnf install net-tools
  netstat -nlp
  On Centos/RHEL:
  sudo yum install net-tools
  netstat -nlp
  Systems which have no netstat utility available can use ss instead. For example to display listening TCP connections:
  `ss -lnt`
  For UDP: `ss -lnu`
  Disabling unnecessary listening services is important to reduce the attack surface of the system.Vendor-supplied defaults and security parameters
- **Vendor-supplied defaults and security parameters**
  PCI section 2.2.3 specifically states that SSLv3 and early TLS versions should no longer be used.
  On Apache installations, SSLv2 and SSL v3 should be disabled by adding the SSLProtocol option, specifying which protocols NOT to use. This is done in the Apache configuration file which is /etc/apache2/httpd.conf in Debian flavours like Ubuntu and /etc/httpd/conf/httpd in Red-Hat based distributions.
  On NGINX too, only the newest TLS versions should be enabled by using the *ssl_protocols directive.* This is added to the configuration file (e.g. /etc/nginx/nginx.conf), and applied to the *http* context. See more details in the nginx SSL module.
  ssl_protocols TLSv1.2 TLSv1.3;
- **Vulnerability management**
  Requirement 5 emphasizes the use of software that is capable of detecting, removing and protectig against all known types of malicious software, and that for systems that are not considered to be susceptible to malware, there should be periodic evaluations to confirm that they continue not to be susceptible.
  To ensure this item of the PCI DSS compliance verification is completely done, the following steps are done:
  To keep ClamAV updated, the utility known as **freshclam i**s used. Additionally monitoring can be set to check that signatures are regularly updated. One option is to use the **clamconf** utility and determine the date of the signatures.

Determine how freshclam is running (freshclam in daemon mode or manual). Additionally check if it is properly logging to **/var/log/clamav/freshclam.log**. Determine if freshclam encountered any issues, like outdated definitions.
ClamAV runs in either daemon mode or as a manual scanner.
Processes can interface with the Clam daemon and request a file descriptor to be tested. The daemon then will return if the file is OK, or a different code (infected, no permissions etc).The manual scanner utility, clamscan can be used to manually check files. This is also a great way of running a regular scan on Linux machines, by using clamscan in a cronjob.
Check if clamscan is scheduled via a cronjob. Additionally check if clamd is running and available for other software components to use it (e.g. mailbox scanning via MTA).
*Logs are to be retained in accordance with PCI DSS Requirement 10.7, h*ence log files have to be checked in /var/log/clamav and the configuration files (/etc/clamav/clamd.conf and /etc/clamav/freshclam.conf) should have a log file defined. The **clamconf** utility can also be used to quickly determine if logging is enabled by running:
clamconf | grep log
PCI DSS 5.3 states that anti-virus configurationsa are to be examined, including the master installation of the software and a sample of system components, to verify that the anti-virus software cannot be disabled or altered by users. Hence the config files should be held only by priviledged users.

- ***Requirement 8*(Authentication)**:
  Systems need to be maintained, which requires legitimate users to have access to them. Within the PCI DSS requirements, there are several controls which highlight the need for proper access, protection, and logging changes. This includes system configuration files (e.g. to PAM or SSH), but also to the logging itself. In other words, we should take appropriate measures to safeguard these configurations itself as well.
  PCI describes in section 8.1.4 that accounts older than 90 days and are unused, should be removed. Unused or inactive accounts on the system might be an unneeded security risk.
  To determine the last time a user logged in, the *last* command can be used.
  Information is stored in binary files **/var/log/wtmp** or rotated files like **/var/log/wtmp.1**.
  Requirement 10.7 specifies that information should be stored/logged for at least 3 months. Enough copies are to be stored on the system itself, or be availed on a central logging server.
  PAM or Pluggable Authentication Modules and SELinux can be combined to ensure the Linux authentication process is highly robust.
  SELinux automatically tries to log and do audits, using the audit subsystem when available or falling back to the regular system logging. For instance, SELinux access vector cache(AVC) denials are logged by default. Denials will be shown in the audit log file *var*log/udit/audit.log but this can be configured explicitly in the /*etc/*audit/auditd.conf file.
  Two-factor authentication can also be achieved with PAM. Depending on the solution, the related module needs to be installed, configured and tested. Modules include pam_google_authenticator.so.
- ***Requirement 10:***
  PCI section 10.7 covers the need for an audit trail. On Linux we have normal logging, audit events. Normal logging is done using syslog which stores varying information such as boot information and kernel events. Of critical importance to PCI compliance is proper log rotation, without destroying(removing or overwritting) previously stored data. The systemd utility, journald which is a journal logging utility will output logs that are of importance to us.
  By default, most log files on Linux based systems will be stored in /var/log. We can do a quick\ check for any files which are world readable, by using *find*.
  ```
  find /var/log -perm -o=r ! -type l
  ```

It is easy to restrict log file viewing of these entries by changing file permissions using chmod.
We need to enable remote audit logging(to centralize audit events from multiple hosts) and we can do this by either enabling syslog forwarding or enabling the audispremote plugin.
This is an SELinux feature. Hence the file */etc/audisp/plugins.d/syslog.conf* will have its active setting set to yes.
System loggers will not be used however, because the use unencrypted and often not even guarenteed, data delivery. With the audisp-remote plugin, audit events can be sent encrypted and with guaranteed delivery.

-