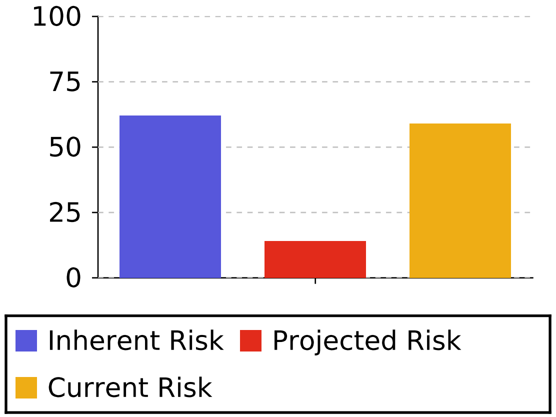
Product Risk Analysis Summary Report

Product Name: 3 Tier Web App
Ref: 3-tier-web-app
Business Units:
Value: Critical
Date: June 10 2018 04:10 PM
Description: A 3 tier web application composed of a Web UI, Web Service and Database

---

Current Risk Summary



Description	Risk Rating
The Inherent Risk before countermeasures were applied	High
The Projected Risk is the level of risk that would be reached should the required countermeasures be implemented	Low
The Current Risk is based on the current implementation status of the countermeasures and test results	High

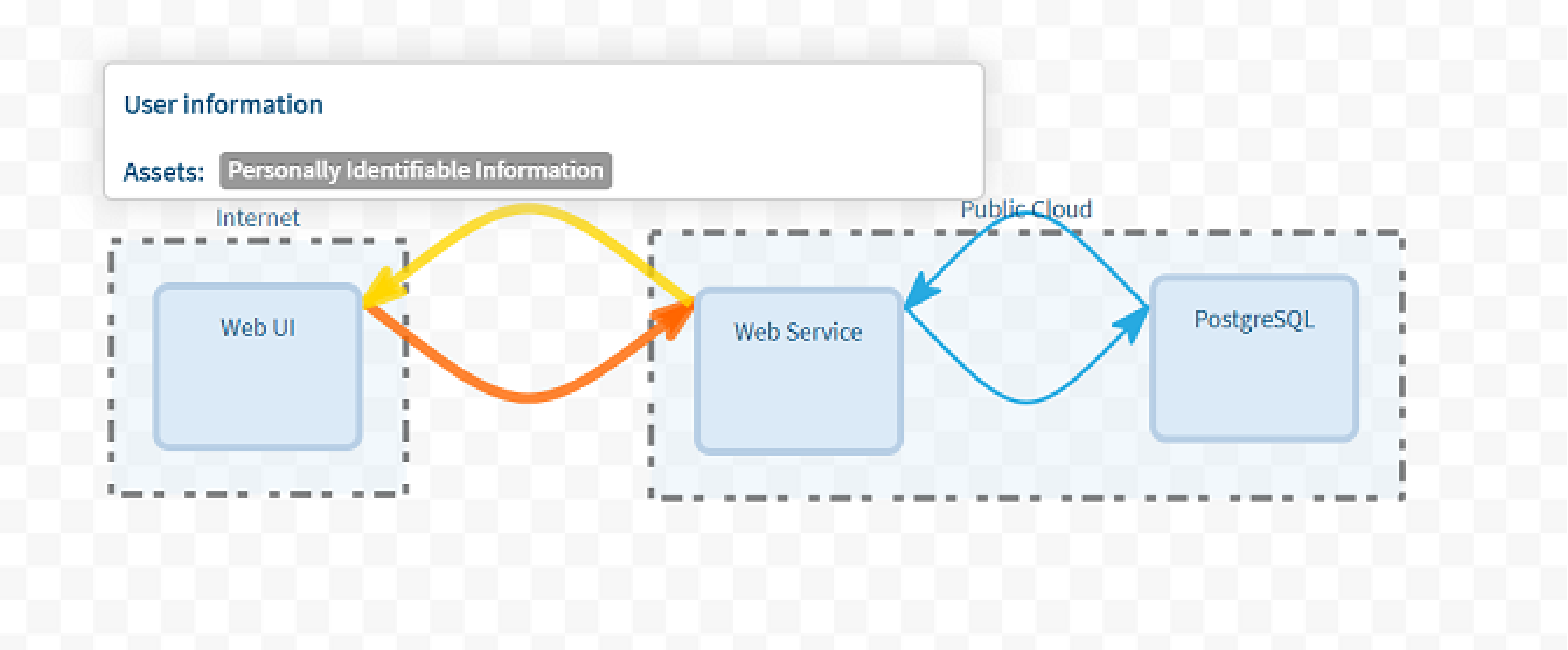
Architecture

Component	Question	Answer
PostgreSQL		
	Which trust zone does the component belong to?	Public Cloud
	Customer Data	Stored
	Select the Data Stores	Data store
	Personally Identifiable Information	Received by component
	Select the SQL Data Store	PostgreSQL
	How will the component be deployed?	Public Cloud
	Which public cloud services?	Amazon Web Services
	Which AWS Services?	Relational Database Service - RDS
	How will data be encrypted between client and server?	Encrypted communication, e.g. HTTPS, SSL/TLS, etc.
	How will the encryption be applied?	At the transport level, e.g. SSL/TLS applied to the whole connection
Web Service		
	Select the server side components	Web Service
	Which trust zone does the component belong to?	Public Cloud
	Customer Data	Processed
		Sent from component
		Received by component
	Personally Identifiable Information	Sent from component
		Received by component
	What technologies are used for the server?	Java Web Container
	Will this server side component implement an authentication function?	Yes, a new authentication function will be created
	How will users authenticate?	Username and Password
	How will the component be deployed?	Public Cloud
	Which public cloud services?	Amazon Web Services
	Which AWS Services?	Elastic Compute Cloud - EC2

Component	Question	Answer
	How will session management between the client and the server side be implemented?	A new session management system will be built as part of this application
	How will the session state be managed?	Using a unique session ID value thats transmitted between the client and server
	How will data be encrypted between client and server?	Encrypted communication, e.g. HTTPS, SSL/TLS, etc.
	How will the encryption be applied?	At the transport level, e.g. SSL/TLS applied to the whole connection
	XML processing	The service will accept XML input from the client
Web UI		
	Which trust zone does the component belong to?	Internet
	Select the client side components	Web UI
	Customer Data	Sent from component
		Received by component
	Personally Identifiable Information	Sent from component
		Received by component
	Authentication with a Browser	The browser presents a form that requests password based login
	How will data be encrypted between client and server?	Encrypted communication, e.g. HTTPS, SSL/TLS, etc.
	How will the encryption be applied?	At the transport level, e.g. SSL/TLS applied to the whole connection
	Which HTML/JS frameworks are in use?	JQuery

Architecture Diagrams

Filename: diagram.png **Date uploaded:** Jun 10, 2018 4:09:07 PM **Username:** admin



Accepted Risks

Component	Use Case	Threat	Risk Rating	Reason for Risk Acceptance	User	Date
PostgreSQL	Store sensitive data	Sensitive data is compromised if the host itself is compromised	High	Risk transferred to the infrastructure team.	admin	Jun 10, 2018 3:56:30 PM
		Sensitive data is compromised if a backup of the data is compromised	Medium	This risk is accepted by the project team, since backup is handled by an external company.	admin	Jun 10, 2018 3:56:08 PM

Risks Not Applicable

Component	Use Case	Threat	Risk Rating	Reason	User	Date

Current Risks

Component	Use Case	Threat	Inherent Risk	Risk Response	Countermeasures	Countermeasure Status	Current Risk
PostgreSQL	Access sensitive data	Sensitive data is compromised though attacks against SSL/TLS	Medium	Partly-Mitigate	Require cryptographically strong TLS cipher suites	Recommended	Medium
			Medium	Partly-Mitigate	Require cryptographically secure protocols (e.g. TLSv1.2 and above)	Required	Medium
	Access service	Attackers obtain unauthorised access by connecting directly to the service	Medium	Mitigate	Require authentication before presenting restricted data	Required	Medium
		Attackers gain access to unauthorised data by exploiting vulnerabilities in the service	Critical	Expose	Apply required security patches to the service	Implemented	Medium
			Critical	Expose	Restrict access to the service at the network layer to reduce exposure	Recommended	Medium
	Read or Post data	Attackers gain unauthorised access to data and/or systems through SQL Injection attacks	High	Mitigate	Validate all data received from the client side	Required	High
			High	Mitigate	Use prepared statements for all database queries	Required	High
			High	Mitigate	Log and reject all data validation failures	Required	High
Web Service	Access sensitive data	Authenticated attackers could gain unauthorised access to sensitive data	High	Mitigate	Apply authorisation checks to segregate and control access to user data	Required	High
		Anonymous users could gain access to sensitive data	High	Partly-Mitigate	Apply authorisation checks to segregate and control access to user data	Required	High
			High	Partly-Mitigate	Authenticate users (single factor authentication)	Required	High
			High	Partly-Mitigate	Authenticate users (multi- or two-factor authentication [2FA])	Recommended	High
		Attackers gain access to data or services by directly accessing the resources	High	Mitigate	Apply authorisation checks to segregate and control access to user data	Required	High
		Sensitive data is exposed to unauthorised personnel in a pre-production environment	High	Expose	Ensure personal and other sensitive data is not exposed in pre-production environments	Recommended	High
		Sensitive data is compromised through network sniffing attacks	High	Mitigate	Encrypt data between the client and server/service	Required	High
		Attackers obtain unauthorised access by connecting directly to the service	Medium	Mitigate	Require authentication before presenting restricted data	Required	Medium
		Sensitive data is compromised though attacks against SSL/TLS	Medium	Expose	Require cryptographically strong TLS cipher suites	Recommended	Medium

Component	Use Case	Threat	Inherent Risk	Risk Response	Countermeasures	Countermeasure Status	Current Risk
			Medium	Expose	Require cryptographically secure protocols (e.g. TLSv1.2 and above)	Recommended	Medium
	Authentication	Attackers gain access to critical functions by compromising the session ID	High	Partly-Mitigate	Terminate user sessions on the server-side after a logout operation	Recommended	High
			High	Partly-Mitigate	Require additional authentication for sensitive operations / high value transactions	Required	High
		Attackers could compromise users' sessions by compproming the session cookie	High	Mitigate	Session ID's should be transmitted securely	Required	High
			High	Mitigate	Session cookie domain attributes should be restricted to prevent exposure	Required	High
			High	Mitigate	Session cookie path attributes should be restricted to prevent exposure	Required	High
		Attackers gain access to the system using default passwords	High	Mitigate	Remove default credentials and role-based accounts from the application	Required	High
		The session ID, and hence the users' session is compromised through brute force attack	High	Mitigate	Use session management functionality provided by the development framework	Required	High
			High	Mitigate	Session tokens should contain sufficient entropy	Required	High
		Dictionary-based or brute force password attack	High	Partly-Mitigate	The login function should distinguish between upper and lower case passwords	Recommended	High
			High	Partly-Mitigate	Offer a password change facility	Required	High
			High	Partly-Mitigate	Require the use of strong passwords	Required	High
			High	Partly-Mitigate	Authenticate users (multi- or two-factor authentication [2FA])	Recommended	High
		Session ID's and hence user sessions are compromised through network sniffing or man in the middle attacks	High	Mitigate	Set the 'secure' flag (or directive) on sensitive cookies	Required	High
		Attackers gain access to user accounts by accessing the password database	High	Mitigate	Store passwords in unrecoverable form to prevent disclosure	Required	High
		Attackers gain access to user accounts by exploiting flaws in the authentication function	High	Mitigate	Enforce authentication on the server-side	Required	High
			High	Mitigate	Ensure authentication fails securely on the server-side	Required	High
		User accounts compromised through username guessing	Medium	Mitigate	Require the use of strong passwords	Required	Medium
			Medium	Mitigate	Implement application and network rate limiting on the login function	Required	Medium

Component	Use Case	Threat	Inherent Risk	Risk Response	Countermeasures	Countermeasure Status	Current Risk
		Sensitive data is compromised through network access	Medium	Expose	Use Network Access Control Lists (NACLs) for blacklisting	Recommended	Medium
		Authentication credentials compromised through network sniffing	Medium	Mitigate	Encrypt data between the client and server/service	Required	Medium
		Attackers gain access to the systems through direct access	Medium	Expose	Use security groups to block ingress to all ports from 0.0.0.0/0	Recommended	Medium
			Medium	Expose	Restrict all traffic with the default security group	Recommended	Medium
			Medium	Expose	Use security groups to block ingress to port 22 from 0.0.0.0/0	Recommended	Medium
			Medium	Expose	Use security groups to block ingress to port 3389 from 0.0.0.0/0	Recommended	Medium
		Usernames could be enumerated through login responses	Medium	Partly-Mitigate	Ensure application errors do not reveal account status	Required	Medium
			Medium	Partly-Mitigate	Ensure failed login timings do not reveal account status	Recommended	Medium
		Attacks against the authentication system may go undetected	Medium	Partly-Mitigate	Log details of user actions within the system	Recommended	Medium
			Medium	Partly-Mitigate	Use a synchronised time source	Required	Medium
	General	Attackers gain unauthorised access to data or services by exploiting known weaknesses in components, libraries, modules, frameworks, platforms and operating systems	High	Mitigate	Regularly check all components of your project for known vulnerabilities	Required	High
		Attackers gain control of the system through a source code leakage	Medium	Expose	Prevent unauthorised access to source code through the service	Recommended	Medium
	Patching	Attackers gain unauthorised access to data on EC2 instances	Medium	Expose	Maintain a patch policy and patch EC2 systems regularly	Recommended	Medium
	Post data	Attackers gain access to the system through Server Side Code Injection	Critical	Mitigate	Validate all data received from external systems	Required	Critical
	Read or Post data	Functionality could be subverted through mass assignment	High	Mitigate	Use a white-list approach to assign values to variables	Required	High
		Execute arbitrary code on the server by manipulating parameters	High	Partly-Mitigate	Use a library that is not vulnerable to Remote File Inclusion	Recommended	High
			High	Partly-Mitigate	Create a mapping to existing objects	Recommended	High
			High	Partly-Mitigate	Validate all data received from the client side	Required	High
			High	Partly-Mitigate	Log and reject all data validation failures	Required	High

Component	Use Case	Threat	Inherent Risk	Risk Response	Countermeasures	Countermeasure Status	Current Risk
			High	Partly-Mitigate	Validate input parameters to prevent HTTP Parameter Pollution	Required	High
			High	Partly-Mitigate	Verify that all input is limited to an appropriate size limit	Required	High
		Attackers gain control of the connection by doing a Man In The Middle attack	High	Mitigate	Encrypt data between the client and server/service	Required	High
		User data or credentials are compromised through network sniffing or man in the middle attacks	High	Partly-Mitigate	Consider HTTP Public Key Pinning (HPKP)	Required	High
			High	Partly-Mitigate	Encrypt data between the client and server/service	Required	High
			High	Partly-Mitigate	Set the HTTP security header 'Strict-Transport-Security' (HSTS)	Required	High
			High	Partly-Mitigate	Preload application URLs to vendor Strict Transport Security domain lists	Recommended	High
		Access system files through XML related Attacks	Medium	Mitigate	Disable external XML entity references in the processor	Required	Medium
			Medium	Mitigate	Define and enforce secure validation through an XSD or DSD schema on XML input data	Required	Medium
		Unauthorised data could be accessed by manipulating parameters sent to the application	Medium	Partly-Mitigate	Log and reject all data validation failures	Required	Medium
			Medium	Partly-Mitigate	Validate input parameters to prevent HTTP Parameter Pollution	Required	Medium
			Medium	Partly-Mitigate	Validate all data received from the client side	Required	Medium
			Medium	Partly-Mitigate	Verify that all input is limited to an appropriate size limit	Required	Medium
			Medium	Partly-Mitigate	Avoid using direct references to files	Required	Medium
			Medium	Partly-Mitigate	Disable Server Side Includes	Recommended	Medium
Web UI	Access sensitive data	Sensitive data is compromised through query parameters in the URL	Critical	Mitigate	Ensure no sensitive data is sent in the URL	Required	Critical
		Attackers with access to a victim's browser could read locally stored data	Critical	Mitigate	Set Cache-Control headers on sensitive and authenticated content	Required	Critical
			Critical	Mitigate	Disable autocompletion of sensitive data	Required	Critical

Component	Use Case	Threat	Inherent Risk	Risk Response	Countermeasures	Countermeasure Status	Current Risk
			Critical	Mitigate	Do not store sensitive data on client side	Required	Critical
			Critical	Mitigate	Clear sensitive and authenticated data from client-side storage	Required	Critical
		Sensitive data is compromised though attacks against SSL/TLS	High	Expose	Require cryptographically strong TLS cipher suites	Implemented	Very Low
			High	Expose	Require cryptographically secure protocols (e.g. TLSv1.2 and above)	Implemented	Very Low
	Authentication	Authentication process is spoofed by replaying login credentials	Critical	Mitigate	Respond to login requests with HTTP 302 redirect	Required	Critical
			Critical	Mitigate	Disable authentication form autocomplete	Required	Critical
		Attackers gain access to in-memory passwords/credentials	Critical	Mitigate	Overwrite passwords, key material, and other secrets maintained in memory when no longer required	Required	Critical
		Attackers with access to a victim's client could read locally stored credentials	Critical	Mitigate	Disable authentication form autocomplete / pre-fill	Required	Critical
		Attackers bypass authentication implemented on the client side	Critical	Expose	Enforce authentication on the server-side	Recommended	Critical
		Authentication credentials posted to a spoofed server	High	Mitigate	Encrypt data between the client and server/service	Required	High
	General	Attackers gain unauthorised access to data or services by accessing a client side secret	Critical	Partly-Mitigate	Implement sensitive logic and data validation on the server-side	Required	Critical
			Critical	Partly-Mitigate	Review code, configuration, and online repositories for secrets and sensitive information systemically	Recommended	Critical
		Attackers gain unauthorised access to data or services by exploiting known weaknesses in components, libraries, modules or frameworks	Critical	Mitigate	Regularly check all components of your project for known vulnerabilities	Required	Critical
	Read or Post data	Attackers could gain access to sensitive data through a too permissive CORS policy	Critical	Mitigate	Restrict Cross Domain Origin policy through HTTP headers	Required	Critical
		Data is posted to a spoofed server	High	Mitigate	Encrypt data between the client and server/service	Required	High
		Attackers could gain access to an open session, if they have access to a user's browser	High	Mitigate	Revoke user sessions after a period of inactivity	Required	High
			High	Mitigate	Revoke user sessions after a fixed time period	Required	High
			High	Mitigate	Provide logout links on all pages that require authentication	Required	High

Component	Use Case	Threat	Inherent Risk	Risk Response	Countermeasures	Countermeasure Status	Current Risk
			High	Mitigate	Terminate user sessions on the server-side	Required	High
		Attackers could gain access to a users' browser through Cross Site Scripting attacks	High	Partly-Mitigate	Perform contextual HTML encoding of all user submitted data	Required	Low
			High	Partly-Mitigate	Set the HTTP security header 'X-Content-Type-Options' from the server	Implemented	Low
			High	Partly-Mitigate	Set the 'httpOnly' flag (or directive) on session cookies	Implemented	Low
			High	Partly-Mitigate	Define a restrictive 'least privilege' Content Security Policy	Required	Low
			High	Partly-Mitigate	Set the HTTP security header 'X-XSS-Protection' from the server	Implemented	Low
	Transaction Authentication	Attackers could cause users to perform specific actions on their behalf through Cross Site Request Forgery attacks	Critical	Partly-Mitigate	Set and verify one time 'Anti CSRF' tokens for sensitive operations	Required	Critical
			Critical	Partly-Mitigate	Require re-authentication for critical operations	Recommended	Critical
		Attackers cause users to peform arbitrary clicks on the site through ClickJacking attacks	Critical	Mitigate	Set the HTTP security header 'X-Frame-Options header' from the server	Required	Critical