PSAM IT Division

*PSAM - Reminder 365*

*Application Security*

*Design Specification*

*0.1*

|  |  |
| --- | --- |
| Title | PSAM Reminder 365 |
| Subject | Application Security Design Specifications |
| Version | 0.1 |

© 2014 PSA Corporation Limited. All rights reserved.

1. **Reference to Master Document**

|  |  |
| --- | --- |
| File server | Document8 |
| URL |  |

1. **Reviewers and approvers**

|  |  |  |
| --- | --- | --- |
| Reviewed by: | | |
| Full name/Designation/Company | Signature | Date |
| <Project Manager> |  |  |
| <Architecture Reviewer> |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Approved by: | | |
| Full name/Designation/Company | Signature | Date |
|  |  |  |
|  |  |  |

1. **Version History**

|  |  |  |
| --- | --- | --- |
| Version | By/When | Summary |
| 0.1 | Venkatesh N |  |
|  |  |  |

♦ *End of Document Control Record* ♦

Contents

[1. Introduction 5](#_Toc502837830)

[2. References 5](#_Toc502837831)

[3. Application Security 6](#_Toc502837832)

[3.1. Authentication 6](#_Toc502837833)

[3.1.1. User ID and Password 6](#_Toc502837834)

[3.1.2. Login 8](#_Toc502837836)

[3.2. Authorisation 9](#_Toc502837954)

[3.2.1. Role-based Access Control Matrix 9](#_Toc502837955)

[3.2.2. Role Definition Database Access Matrix 12](#_Toc502837956)

[3.2.3. Role Definition Hierarchy Design 12](#_Toc502837957)

[3.3. Administration 12](#_Toc502838056)

[3.3.1. Best Practices Checklists 12](#_Toc502838057)

[3.3.2. Other Administration Requirements and Designs 13](#_Toc502838058)

[3.4. Audit 13](#_Toc502838059)

[3.5. Web Application Security 15](#_Toc502838060)

[3.5.1. Secure Interaction Between Components 15](#_Toc502838061)

[3.5.2. Proper Resource Management 19](#_Toc502838062)

[3.5.3. Proper Defences 22](#_Toc502838063)

[3.5.4. Other security considerations. 24](#_Toc502838064)

[4. System Security 26](#_Toc502838065)

[5. LDAP Access 27](#_Toc502838066)

|  |
| --- |
| **Note:**   1. This document should be prepared, based on Use Case Specification (UCS), and the design requirement gathered in Technical Requirements and Design Spec (TRDS) and UML Model (Detailed Design Specifications using ROSE UML Implementation). 2. It contains the detailed design specifications on the aspect of application security, gathered and exploded in the TRDS. It covers the security aspect of an application, in the following areas: Authentication, Authorization, Administration, Audit and Single-sign on. Specific sections are reserved for Community module where the security settings and usage of it need to be reviewed. 3. The objective of this document is to ensure that the project takes into consideration of security during design phase. It also served to remind the project to adhere to the ITD Governance on Application Security Policies and Guidelines document. 4. Please note that those checklists prefixed with a ‘\*’ is a security best practices or guidelines. **Those that without a** ‘\*’ **in their prefix is a security policy that cannot be deviated**. Any deviation from the security policies would require approval from ITD Security Steering Committee. 5. Please note that the security checklists listed in this document are not exhaustive. In the event that there are some improvements required, please inform SAT. 6. This document must be maintained updated at all times, during the project lifecycle. All amendments (initial and subsequent) must be reviewed and signed-off. |

# Introduction

The objective of this project is to seek a complete proposal to develop a simple and interactive intranet web application that allows users to manage and monitor the expiry dates of equipment, certificates, and contracts. When nearer to date, the application should be smart enough to consolidate the expiring items of the same category and send an email to notify the respective stakeholders of that user group. This web application mainly consists of the following modules:-

* Authentication and Authorization Module
* Equipment Reminder Module (Craft & Store) for 2 main group of users for this module, namely FMD-Operation and FMD-Technical Service. Each group of users has their own equipment to monitor and they cannot manage equipment of the other groups unless being granted the necessary roles.
* Staff Reminder Module for 3 main group of users for this module, namely HR, FMD-Operation and FMD-Technical Service. Each group of users has their own records to monitor and they cannot manage records of the other groups unless being granted the necessary roles.
* Contracts Reminder Module for 3 main group of users for this module, namely FMD, Procurement and IT. Each group of users has their own contract records to monitor.
* Setting/Configuration Module
* Reminder Dashboards

# References

|  |
| --- |
| The following documents have been submitted to PSA or are in work in progress currently.   * User Requirement Specifications (URS) * Use Cases * Technical Requirements & Design Specification * ER Diagrams & Database Design * UI Mock-ups can be accessed from <https://projects.invisionapp.com/share/4RDCWWX7C#/screens>   *http://webdemo.saksoft.com/psa-365/settings.html* |

# Application Security

## Authentication

### User ID and Password

State your user ID and password design, using the following checklists.

Note:

1. For application that uses Community module and Corporate LDAP for their authentication implementation, you may skip this section as they are already complied.
2. If you are using an authentication implementation other than Community module and Corporate LDAP (for e.g. application may be using ERP internal built-in login page), it is **mandatory** to state how the above ‘Yes’ compliance lists are achieved. (If it is too difficult to provide an illustration here, please describe in words instead.)

|  |  |  |
| --- | --- | --- |
| **ID and Password**  **Checklists** | **Comply(Y/N/NA)** | **For non-compliance and not applicable, please provide justification.** |
| **User ID** | | |
| Individual user ID and password are required to access the application. | Y |  |
| Login ID format follows the standard format of NT Login ID. | Y |  |
| **Password** | | |
| Password is masked while being entered. | Y |  |
| Passwords are stored in encrypted form, and access to the password store is strictly controlled. | Y | Managed by Windows AD |
| Passwords must consist of a minimum of 8 alphanumeric characters. | Y | Managed by Windows AD |
| Password dictionary check is implemented. | Y | Managed by Windows AD |
| Password expiry check: Mandatory change of password is enforced once every 90 days for normal users and once every 30 days for privileged users. | Y | Managed by Windows AD |
| Change Password: New password is not the same as the Login ID | Y | Managed by Windows AD |
| Password history: New password is different from existing and 2 previous passwords. | Y | Managed by Windows AD |
| Password Lockout check: Password is lock-out after 4 successive login fail attempts. | Y | This is not being checked at application level. If the Active Directory has provision to lock the message will be shown on the application |
| Password is not captured and printed in any of the application log files. | Y |  |
| Password is not hard-coded in any of the application source codes. | Y |  |
| Password is stored in encrypted form in the user data store. | Y | Managed by Windows AD |
| \* Password is stored in encrypted form when stored in configuration file. | NA | Not storing any password in application |
| Functions are built-in to support the suspension and deletion of inactive IDs. | Y |  |

### Login

Note: This section is mandatory only when you are building your own login page. If you are using the existing common login page available in CITOS/EIP/PORTNET portal, you may skip this section by indicating “N.A” and the reason.

|  |  |  |
| --- | --- | --- |
| **Login Checklists** | **Comply(Y/N/NA)** | **For non-compliance and not applicable, please provide justification.** |
| Display the warning message:  “WARNING: Access to information on this machine and network is restricted to authorised personnel only. Any unauthorised user is subject to criminal prosecution under the Computer Misuse Act (Cap. 50A).” | Y |  |
| Display last successful login date and time after user successfully login    For application authenticating against Corp LDAP, please display the message as “Last successful authentication on Corporate LDAP…” | Y |  |
| Display last unsuccessful login date and time after user successfully login  For application authenticating against Corp LDAP, please display the message as “Last unsuccessful authentication on Corporate LDAP…” | Y |  |
| \* Changing of user password in Corporate/Portnet LDAP can only be done at a centralised portal, community admin module. | Y | Managed by Windows AD |
| Passwords must be transmitted over an encrypted channel e.g. over SSL for web portals. | Y |  |
| Valid SSL certificates must be used for login web portals. For intranet web portals, internal Corp CA SSL certificates shall be used. For internet web portals, commercial SSL certificates shall be used. | Y |  |











## Authorisation

### Role-based Access Control Matrix

|  |
| --- |
| Sequence diagram: |

### Role Definition Database Access Matrix

|  |
| --- |
| *What to include:*  To indicate the DB operations that an application role can perform.  For example, a guest user can only perform read DB operation while an admin user can perform a read/create/update/delete operation. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Read | Create | Update | Delete |
| Group Admin | ☒ | ☒ | ☒ | ☒ |
| HR Admin | ☒ | ☒ | ☒ | ☒ |
| IT Admin | ☒ | ☒ | ☒ | ☒ |
| Reminder Module Users\* | ☒ | ☒ | ☒ | ☒ |

\* Access privileges for these users can be modified at application level

### Role Definition Hierarchy Design

Level 1 - Group, HR and IT Admin

Level 2 - Reminder Module Users











## Administration

### Best Practices Checklists

Note:

1. If you do NOT have any user or company administration requirement, or your application will be using Corporate/PORTNET community administration processes, you may skip this section by indicating “N.A” and the reason.
2. If you have user or company administration requirements, but are not leveraging on the existing Corporate/PORTNET established community administration processes, please go through the checklists below and indicate how each activity is realized.

|  |  |
| --- | --- |
| **Administration Checklists** | **Please state how it will be realised. If not applicable, please state reasons.** |
| **User Administration** | |
| \* Creation of Account | Y |
| \* Maintenance of account (such as update/delete user account, change password, assignment of role)  If there is delegated administration, please state how the delegated administrator account is handled. | Y |
| \* Maintenance of Password (re-issue new password, unlock password, etc) | Managed by Windows AD |
| \* Housekeeping of in-active accounts (to un-assign role if user no longer needs the access) | Provide SQL scripts to remove inactive user |
| \* Procedures for end user to apply for new account and inform of changes. | Managed by Windows AD |
| \* Procedure for dissemination of password | Using Windows AD Account |
| **Company Administration (for application that has external user)** | |
| \* Creation of Company Account | NA |
| \* Maintenance of Company account | NA |
| \* Maintenance of product (subscribe  and unsubscribe product) | NA |
| \* User defined roles (aggregate of default roles from various product) | NA |

### Other Administration Requirements and Designs

|  |
| --- |
| *What to include:*  Indicate additional administration requirements and designs, if it is not covered in the checklists.  Please have a sub section for each administration requirement and the proposed design. |

## Audit

|  |
| --- |
| *What to include:*  Indicate the design of the audit trail facility in your application. An audit trail facility here is for the purpose of security review and audit of user access.  Note: Application leveraging on Corporate or Portnet Community administration processes, or Siteminder security framework, would have satisfied all the security-related audit checklists below. |

|  |  |  |
| --- | --- | --- |
| **Security Audit Implementation Checklists** | **Comply(Y/N/NA)** | **For non-compliance and not applicable, please provide justification.** |
| **Security Aspect** |  |  |
| Audit trails are created for both success and fail login events. There should be a trace of login/logout, access violations. | Y |  |
| OS-, database- and application- audit trails shall contain accounted and timestamped log of commands executed at that level by administrative accounts (i.e. who executed what at when). | Y |  |
| OS logs shall be piped to a centralised syslog server. Detailed steps kindly refer to: <http://confluence.psa:8008/download/attachments/17784/Procedures+for+the+review+of+third-party+monitored+logs.pdf?version=1> | Y |  |
| Access to sensitive pages or critical transactions should also be logged. This may involve consultation with business user. | Y |  |
| The information to log should contain at least: date, time, source information (such as IP address), user ID, department/organisation ID, action performed and status. | Y |  |
| Audit information is kept for the specified period as per security and business requirement.  Login/security related for 3 months.  Finance related for a minimum of 5 years. | Y |  |
| Access to audit trail logs must be restricted. | Y |  |
| Have facility to periodically review the security log for any access violation. | Y |  |
| Transactional system shall have an audit trial log, where changes to business information are recorded. | Y |  |
| The operating system shall complete a server security review assessed by IT Security Team. Refer to “Server Review” section under <http://confluence.psa:8008/display/itsec/ITI+Security> for detailed instructions. | Y |  |

## Web Application Security

|  |
| --- |
| *What to include:*  For web applications, below are checklists to protect against the top 25 most dangerous programming errors, in particular for web application coding. |

### Secure Interaction Between Components

|  |  |  |
| --- | --- | --- |
| **Checklists** | **Comply(Y/N/NA)** | **For non-compliance and not applicable, please provide justification.** |
| **Proper Input Validation**  (<http://cwe.mitre.org/top25/#CWE-20>) | | |
| Input is validated when invoking code that crosses language boundaries, such as from an interpreted language to native code.  For example, even though Java may not be susceptible to buffer overflows, providing a large argument in a call to native code might trigger an overflow. | NA | No code is crossing language boundaries |
| Input type is directly converted into expected data type, such as using a conversion function that translates a string into a number  After conversion, input's values are ensured to fall within the expected range of allowable values and multi-field consistencies are maintained. | Y |  |
| Inputs are decoded to application's internal representation before being validated. (CWE-180, CWE-181).  Application does not inadvertently decode the same input twice (CWE-174). | Y |  |
| **Proper Encoding or Escaping of Output**  **(**<http://cwe.mitre.org/top25/#CWE-116>) | | |
| During data exchange between components, components use same character encoding. | Y |  |
| **Successful Preservation of SQL Query Structure**  **(**<http://cwe.mitre.org/top25/#CWE-89>) | | |
| When dynamically-generated query strings are needed, proper encoding and escaping of inputs are used.  For example, the Oracle DBMS\_ASSERT package can check or enforce that parameters have certain properties that make them less vulnerable to SQL injection.  For MySQL, the mysql\_real\_escape\_string() API function is available in both C and PHP. | Y |  |
| A whitelist of acceptable inputs is used. Any input that does not strictly conform to specifications is rejected or transformed into something that does.  A blacklist is used to reject any unexpected inputs and detect potential attacks. | Y |  |
| A standard input validation mechanism is used to validate all input for length, type, syntax, and business rules before being accepted for further processing. | Y |  |
| When SQL query strings are constructed, use stringent whitelists that limit the character set based on the expected value of the parameter in the request. | Y |  |
| Proper output encoding, escaping, and quoting is implemented.  For example, name "O'Reilly" would pass validation, since it is a common name in English language. However, it cannot be directly inserted into the database because it contains the "'" apostrophe character. This character would need to be escaped or otherwise handled. In this case, stripping the apostrophe might reduce the risk of SQL injection, but it would produce incorrect behaviour because the wrong name would be recorded. | Y |  |
| When feasible, Meta-characters are disallowed entirely, instead of escaping them.  For example, after data is entered into the database, later processes may neglect to escape meta-characters before use, and may not have control over those processes. | Y |  |
| **Successful Preservation of Web Page Structure to protect against Cross-Site Scripting (**<http://cwe.mitre.org/top25/#CWE-79>) | | |
| For any data that will be output to another web page, especially any data that was received from external inputs, appropriate encoding is used on all non-alphanumeric characters. | Y |  |
| A strong character encoding such as ISO-8859-1 or UTF-89 is used. | Y |  |
| With Struts, all data from form beans are written with the bean’s filter attribute set to true. | NA | Spring MVC is used |
| Session cookie is set to HttpOnly.  In browsers that support the HttpOnly feature, this attribute can prevent the user’s session cookie from being accessible to malicious client-side scripts that use document.cookie. | Y |  |
| When dynamically constructing web pages, stringent whitelists that limit the character set based on expected value of the parameter in the request are used.  All input is validated and cleansed, not just parameters that the user is supposed to specify, but all data in the request, including hidden fields, cookies, headers, the URL itself, etc. | Y |  |
| **Successful preservation of OS Command Structure to protect against OS Command Injection**  **(**<http://cwe.mitre.org/top25/#CWE-78>) | | |
| Arguments are properly quoted and special characters within those arguments properly escaped.  If special characters are needed, arguments are wrapped in quotes, and all other characters that do not pass a strict whitelist are escaped. | Y |  |
| If program to be executed allows arguments to be specified within an input file or from standard input, then input file is used to pass arguments instead of command line. | Y |  |
| **Secured Transmission of Sensitive Information**  **(**http://cwe.mitre.org/top25/#CWE-319) | | |
| When using web applications with SSL, SSL is used for the entire session from login to logout, not just for initial login page. | Y |  |
| **Protection against Cross-site request forgery**  **(**<http://cwe.mitre.org/top25/#CWE-352>) | | |
| HTTP referrer header is checked to see if the request originated from an expected page. | Y |  |
| **Protection against Race Condition**  **(**<http://cwe.mitre.org/top25/#CWE-362>) | | |
| When using multi-threading, only thread-safe functions on shared variables are used. | Y |  |
| Atomic operations on shared variables are used. | Y |  |
| mutex is used if available. | Y |  |
| Double-checked locking and other implementation errors that arise when trying to avoid the overhead of synchronization are avoided. | Y |  |
| Interrupts or signals over critical parts of the code are disabled, while ensuring that the code does not go into a large or infinite loop. | Y |  |
| Volatile type modifier for critical variables are used to avoid unexpected compiler optimization or reordering. | Y |  |
| **Protection against Error Message Information Leak**  **(**<http://cwe.mitre.org/top25/#CWE-209>) | | |
| Exceptions are handled internally and do not display errors containing potentially sensitive information to a user. | Y |  |

### Proper Resource Management

|  |  |  |
| --- | --- | --- |
| **Issue** | **Comply(Y/N/NA)** | **For non-compliance and not applicable, please provide justification.** |
| **Successful limiting of operations within the bounds of a memory buffer.**  (<http://cwe.mitre.org/top25/#CWE-119>) | | |
| Buffer is double-checked to be as large as specified. | Y |  |
| Buffer boundaries are checked to ensure that calling functions such as strncpy() are not in danger of writing past allocated space.  Where feasible, all input strings are truncated to a reasonable length before passing them to the copy and concatenation functions. | Y |  |
| **External Control of Critical State Data**  (<http://cwe.mitre.org/top25/#CWE-642>) | | |
| If using PHP, application is configured so that it does not use register\_globals. | NA | Using Java |
| **External Control of File Name or Path**  (<http://cwe.mitre.org/top25/#CWE-73>) | | |
| For filenames, stringent whitelists are used that limit the character set to be used. Where feasible, only a single “.” Character in the filename is allowed and directory separators such as “/” are excluded.  Whitelist of allowable file extensions should be used. | Y |  |
| Built-in path canonicalization function (such as realpath() in C) that produces the canonical version of the pathname, which effectively removes “..” sequences and symbolic links should be used. | Y |  |
| OS-level permissions are used and a low-privileged user is used to run the application so as to limit the scope of any successful attack. | Y |  |
| **Trusted Search Path**  (<http://cwe.mitre.org/top25/#CWE-426>) | | |
| When invoking other programs, fully-qualified pathnames is used. | Y |  |
| Environment is sanitized before invoking other programs.  This includes the PATH environment variable, LD\_LIBRARY\_PATH and other settings that identify the location of code libraries, and any application-specific search paths. | Y |  |
| Search path is checked before use and any elements that are likely to be unsafe are removed, such as the current working directory or temporary files directory. | Y |  |
| Readily available functions that require explicit paths are used.  For example, system() in C does not require a full path since the shell can take care of it, while execl() and execv() require a full path. | Y |  |
| **Proper Resource Shutdown or Release**  (<http://cwe.mitre.org/top25/#CWE-404>) | | |
| Memory is freed at all exit points for function including error conditions. | Y | This is handled by Java |
| Memory is allocated/freed using matching functions such as malloc/free, new/delete, and new[]/delete[]. | Y |  |
| When releasing complex object or structure, all member components are disposed, not just the object itself. | Y |  |
| **Proper Initialization**  (<http://cwe.mitre.org/top25/#CWE-665>) | | |
| All variables and data stores, during declaration or just before first usage, are explicitly initialized. | Y |  |
| Software with settings that generate warnings about uninitialized variables or data is run or compiled. | Y |  |
| **Proper Calculation**  (<http://cwe.mitre.org/top25/#CWE-682>) | | |
| Input validation is performed on any numeric inputs, ensuring that they are within the expected range. | Y |  |
| Appropriate type is used for the desired action.  For example, in C/C++, only use unsigned types for values that could never be negative. | Y |  |
| Languages, libraries, or frameworks that make it easier to handle numbers without unexpected consequences are used. | Y |  |

### Proper Defences

|  |  |  |
| --- | --- | --- |
| **Issue** | **Comply(Y/N/NA)** | **For non-compliance and not applicable, please provide justification.** |
| **Proper Access Control (Authorization)**  (<http://cwe.mitre.org/top25/#CWE-285>) | | |
| All pages containing sensitive information are not cached.  All such pages restrict access to requests that are accompanied by an active and authenticated session token associated with a user who has the required permissions to access that page. | Y |  |
| **Use of Strong Cryptographic Algorithm**  (<http://cwe.mitre.org/top25/#CWE-327>) | | |
| Weak cryptographic algorithms such as MD4, MD5, DES are not used. Instead, strong algorithms such as AES and DES3 are used. | Y |  |
| Cryptographic keys are properly managed and protected. | Y |  |
| **Strong Authentication.**  (<http://cwe.mitre.org/top25/#CWE-259>) | | |
| For outbound authentication, passwords are stored outside the code in a strongly-protected, encrypted configuration file or database that is protected from access by all outsiders, including other local users on the same system.  If cannot use encryption to protect the file, permissions are set as restrictive as possible.  Keys are also properly protected. | Y |  |
| For inbound authentication, default username and password for first time logins is not hardcoded.  Instead, a “first login” mode is enabled which requires the user to enter a unique strong password. | Y |  |
| Access to features that require hard-coded password is restricted.  For example, a feature might only be enabled through the system console instead of through a network connection. | NA | No requirement on such feature |
| For inbound authentication, only strong one-way hashes to passwords are used.  These hashes are stored in configuration file or database with appropriate access control.  When handling an incoming password during authentication, hash of the password is compared to hash that was saved. | NA | Authentication is done via Windows AD credential |
| Salts are randomly assigned for each separate hash generated. | Y |  |
| Network messages sent are tagged and checksummed with time-sensitive values so as to prevent replay attacks. | Y |  |
| **Use of Sufficiently Random Values**  (<http://cwe.mitre.org/top25/#CWE-330>) | | |
| If pseudo-random number generator is used, a seed of at least 256-bit is used to produce a “random enough” number. | NA |  |
| **Execution with Appropriate Privileges**  (<http://cwe.mitre.org/top25/#CWE-250>) | | |
| Privileges are dropped as soon as possible after use.  Privileges are to be ensured to have been dropped successfully. | Y |  |
| Where circumstances require extra privileges to be run, minimum access level necessary to allow only these actions is ensured. | Y |  |

### Other security considerations.

|  |  |  |
| --- | --- | --- |
| **Checklists** | **Comply(Y/N/NA)** | **For non-compliance and not applicable, please provide justification.** |
| Automatic timeout/logout after 15 minutes of inactivity is enforced for sensitive systems. | Y |  |
| \* No hard coding of IP address in the application. | Y |  |
| No confidential information (such as password) is sent to external parties through email. | Y |  |
| Checks are implemented to ensure that integrity of data is maintained when they are being passed between systems. | Y |  |
| **Single-sign-on** |  |  |
| Sensitive sites require higher level protection and therefore an additional challenge for access. | NA |  |
| The user session (usually kept in cookies) must be properly disposed of when it is no longer needed. | NA |  |
| Single sign-on session time-out must be enforced. 30 min recommended. (Not to be confused with application session time-out) | NA |  |
| For sensitive sites, only secure cookies are used to keep session state. This means the cookie will not be sent over HTTP where they will be in the clear. | NA |  |
| Persistent cookies cannot be used to maintain SSO sessions for multiple browser sessions since the cookie will be written to the client harddisk. | NA |  |
| For web based applications, users are denied access to the server directory listing via the browser. | Y |  |
| For Internet-facing web applications, the robots.txt file shall be created to ensure that Internet search engines (e.g. google) do not crawl the website to harvest information not intended for this purpose. | NA |  |
| Where email addresses are publicly accessible, it shall be obfuscated such that malicious email harvesting cannot succeed. | NA |  |

# System Security

|  |
| --- |
| *What to include:*  This section concerns IT security from the system perspective, where access to, or usage on system resources that may impose security hazards.  Please go through the System Security checklists using the table below. |

|  |  |  |
| --- | --- | --- |
| **System Security Checks** | **Comply(Y/N/NA)** | **For non-compliance and not applicable, please provide justification.** |
| \* Internet, extranet and intranet applications are running on separate servers, and there is no direct connection between them. | Y |  |
| The operating system shall complete a server security review. Any issues reported in the review results shall be reported to the IT Security Team.  Refer to “Server Review” section under <http://confluence.psa:8008/display/itsec/ITI+Security> for detailed instructions. | Y |  |
| The database shall complete a database security review based on latest database security deployment checklist.  Any issues reported in the review results shall be reported to the IT Security Team. | Y |  |

# LDAP Access

|  |
| --- |
| For application that perform custom search outside the use of Community framework.  This is to ensure that any search perform against Corporate/Portnet LDAP is done in a proper way, so as to prevent application conducting resource-intensive searches indiscriminately.  What to include:  Developers have to declare all the custom searches conducted against Corporate/Portnet LDAP.  Fill in the following information in the table:  - Base DN where the search starts from.  - Search filter.  - Attributes to be retrieved.  - Justification for performing this custom search |

Example:

|  |  |  |  |
| --- | --- | --- | --- |
| **Base DN** | **Search Filter** | **Attributes** | **Justification** |
| ou=psa, ou=Customers, l=SGSIN, o=portnet.com | (objectclass=XYZ) | All |  |
| ou=psa, ou=Customers, l=SGSIN, o=portnet.com | (uid=’ABC’) | All |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Base DN** | **Search Filter** | **Attributes** | **Justification** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

♦ *End of Document* ♦