the CENTER for INTERNET SECURITY

Security Configuration Benchmark For

# Sybase ASE 15.0

Version 1.0.0
September 2009


http://cisecurity.org
feedback@cisecurity.org

**Background.**

CIS provides benchmarks, scoring tools, software, data, information, suggestions, ideas, and other services and materials from the CIS website or elsewhere ("**Products**") as a public service to Internet users worldwide. Recommendations contained in the Products ("**Recommendations**") result from a consensus-building process that involves many security experts and are generally generic in nature. The Recommendations are intended to provide helpful information to organizations attempting to evaluate or improve the security of their networks, systems and devices. Proper use of the Recommendations requires careful analysis and adaptation to specific user requirements. The Recommendations are not in any way intended to be a "quick fix" for anyone's information security needs.

**No representations, warranties and covenants**.

CIS makes no representations, warranties or covenants whatsoever as to (i) the positive or negative effect of the Products or the Recommendations on the operation or the security of any particular network, computer system, network device, software, hardware, or any component of any of the foregoing or (ii) the accuracy, reliability, timeliness or completeness of any Product or Recommendation. CIS is providing the Products and the Recommendations "as is" and "as available" without representations, warranties or covenants of any kind.

**User agreements.**

By using the Products and/or the Recommendations, I and/or my organization ("**we**") agree and acknowledge that:

No network, system, device, hardware, software or component can be made fully secure;
We are using the Products and the Recommendations solely at our own risk;

We are not compensating CIS to assume any liabilities associated with our use of the Products or the Recommendations, even risks that result from CIS's negligence or failure to perform;

We have the sole responsibility to evaluate the risks and benefits of the Products and Recommendations to us and to adapt the Products and the Recommendations to our particular circumstances and requirements;

Neither CIS, nor any CIS Party (defined below) has any responsibility to make any corrections, updates, upgrades or bug fixes or to notify us if it chooses at it sole option to do so; and

Neither CIS nor any CIS Party has or will have any liability to us whatsoever (whether based in contract, tort, strict liability or otherwise) for any direct, indirect, incidental, consequential, or special damages (including without limitation loss of profits, loss of sales, loss of or damage to reputation, loss of customers, loss of software, data, information or emails, loss of privacy, loss of use of any computer or other equipment, business interruption, wasted management or other staff resources or claims of any kind against us from third parties) arising out of or in any way connected with our use of or our inability to use any of the Products or Recommendations (even if CIS has been advised of the possibility of such damages), including without limitation any liability associated with infringement of intellectual property, defects, bugs, errors, omissions, viruses, worms, backdoors, Trojan horses or other harmful items.

**Grant of limited rights.**

CIS hereby grants each user the following rights, but only so long as the user complies with all of the terms of these Agreed Terms of Use:

Except to the extent that we may have received additional authorization pursuant to a written agreement with CIS, each user may download, install and use each of the Products on a single computer;

Each user may print one or more copies of any Product or any component of a Product that is in a .txt, .pdf, .doc, .mcw, or .rtf format, provided that all such copies are printed in full and are kept intact, including without limitation the text of this Agreed Terms of Use in its entirety.

**Retention of intellectual property rights; limitations on distribution.**

The Products are protected by copyright and other intellectual property laws and by international treaties. We acknowledge and agree that we are not acquiring title to any intellectual property rights in the Products and that full title and all ownership rights to the Products will remain the exclusive property of CIS or CIS Parties. CIS reserves all rights not expressly granted to users in the preceding section entitled "Grant of limited rights." Subject to the paragraph entitled "Special Rules" (which includes a waiver, granted to some classes of CIS Members, of certain limitations in this paragraph), and except as we may have otherwise agreed in a written agreement with CIS, we agree that we will not (i) decompile, disassemble, reverse engineer, or otherwise attempt to derive the source code for any software Product that is not already in the form of source code; (ii) distribute, redistribute, encumber, sell, rent, lease, lend, sublicense, or otherwise transfer or exploit rights to any Product or any component of a Product; (iii) post any Product or any component of a Product on any website, bulletin board, ftp server, newsgroup, or other similar mechanism or device, without regard to whether such mechanism or device is internal or external, (iv) remove or alter trademark, logo, copyright or other proprietary notices, legends, symbols or labels in any Product or any component of a Product; (v) remove these Agreed Terms of Use from, or alter these Agreed Terms of Use as they appear in, any Product or any component of a Product; (vi) use any Product or any component of a Product with any derivative works based directly on a Product or any component of a Product; (vii) use any Product or any component of a Product with other products or applications that are directly and specifically dependent on such Product or any component for any part of their functionality, or (viii) represent or claim a particular level of compliance with a CIS Benchmark, scoring tool or other Product. We will not facilitate or otherwise aid other individuals or entities in any of the activities listed in this paragraph.

We hereby agree to indemnify, defend and hold CIS and all of its officers, directors, members, contributors, employees, authors, developers, agents, affiliates, licensors, information and service providers, software suppliers, hardware suppliers, and all other persons who aided CIS in the creation, development or maintenance of the Products or Recommendations ("**CIS Parties**") harmless from and against any and all liability, losses, costs and expenses (including attorneys' fees and court costs) incurred by CIS or any CIS Party in connection with any claim arising out of any violation by us of the preceding paragraph, including without limitation CIS's right, at our expense, to assume the exclusive defense and control of any matter subject to this indemnification, and in such case, we agree to cooperate with CIS in its defense of such claim. We further agree that all CIS Parties are third-party beneficiaries of our undertakings in these Agreed Terms of Use.

**Special rules.**

CIS has created and will from time to time create special rules for its members and for other persons and organizations with which CIS has a written contractual relationship. Those special rules will override and supersede these Agreed Terms of Use with respect to the users who are covered by the special rules. CIS hereby grants each CIS Security Consulting or Software Vendor Member and each CIS Organizational User Member, but only so long as such Member remains in good standing with CIS and complies with all of the terms of these Agreed Terms of Use, the right to distribute the Products and Recommendations within such Member's own organization, whether by manual or electronic means. Each such Member acknowledges and agrees that the foregoing grant is subject to the terms of such Member's membership arrangement with CIS and may, therefore, be modified or terminated by CIS at any time.

**Choice of law; jurisdiction; venue.**

We acknowledge and agree that these Agreed Terms of Use will be governed by and construed in accordance with the laws of the State of Maryland, that any action at law or in equity arising out of or relating to these Agreed Terms of Use shall be filed only in the courts located in the State of Maryland, that we hereby consent and submit to the personal jurisdiction of such courts for the purposes of litigating any such action. If any of these Agreed Terms of Use shall be determined to be unlawful, void, or for any reason unenforceable, then such terms shall be deemed severable and shall not affect the validity and enforceability of any remaining provisions. We acknowledge and agree that we have read these Agreed Terms of Use in their entirety, understand them and agree to be bound by them in all respects.

# Table of Contents

# Overview

This guide, *Security Configuration Benchmark for Sybase ASE 15.0.x,* provides security configuration guidance for Sybase Adaptive Server Enterprise (ASE). Sybase ASE is an enterprise-level relational database management system (RDBMS) with support for Java, file system access and XML services. The following versions of Sybase ASE are in scope in this guide:

- Sybase ASE 15.0
- Sybase ASE 15.0.1
- Sybase ASE 15.0.2

Unless explicitly specified, when the server is referred to simply as "Sybase ASE", it should be understood that the setting applies to all version in scope.

The configuration settings in this document represent security best practices for the above versions only; consequently settings that pertain to remote server interaction should be reviewed carefully in environments that consist of both older and current versions of Sybase ASE.

This guide was tested against Sybase ASE 15.0.2 running on Windows 2003 Server R2 and Fedora Core 10. To obtain the latest version of this guide, please visit http://cisecurity.org. If you have questions, comments, or have identified ways to improve this guide, please write us at feedback@cisecurity.org.

## Consensus Guidance

This guide was created using a consensus review process comprised of volunteer and contract subject matter experts. Consensus participants provide perspective from a diverse set of backgrounds including consulting, software development, audit and compliance, security research, operations, government, and legal.

## Intended Audience

This document is intended for system, database, and application administrators, security specialists, auditors, and help desk personnel who plan to develop, deploy, assess, or secure solutions that incorporate Sybase ASE 15.

## Acknowledgements

The following individuals and organizations have demonstrated a commitment to the IT security community by contributing greatly to the consensus review of this configuration guide:

**Author**
John Heasman, *Next Generation Security Software*

**Contributors and Reviews**
Barbara Banks, *Sybase, Inc.*
Rajnish K. Chitkara, *Sybase, Inc*
Rebecca Heffel, *University of Washington*
Mike de Libero, *MDE Development, LLC*
Blake Frantz, *Center for Internet Security*
Vivek Kandiyanallur, *Sybase, Inc.*
Alan Madsen, *Sybase, Inc.*
Christian Monberg, *Hornall Anderson, Inc.*
Steven Piliero, *Center for Internet Security*
Chad Thunberg, *Leviathan Security Group, Inc.*

## Typographic Conventions

The following typographical conventions are used throughout this guide:

| Convention | Meaning |
|---|---|
| `Stylized Monospace font` | Used for blocks of code, command, and script examples. Text should be interpreted exactly as presented. |
| `Monospace font` | Used for inline code, commands, or examples. Text should be interpreted exactly as presented. |
| *<italic font in brackets>* | Italic texts set in angle brackets denote a variable requiring substitution for a real value. |
| *Italic font* | Used to denote the title of a book, article, or other publication. |
| **Note** | Additional information or caveats |

## Configuration Levels

This section defines the configuration levels that are associated with each benchmark recommendation. Configuration levels represent increasing levels of security assurance.

### *Level-I Benchmark settings/actions*

Level-I Benchmark recommendations are intended to:

- be practical and prudent;
- provide a clear security benefit; and
- do not negatively inhibit the utility of the technology beyond acceptable means

### *Level-II Benchmark settings/actions*

Level-II Benchmark recommendations exhibit one or more of the following characteristics:

- may negatively inhibit the utility or performance of the technology
- acts as defense-in-depth measure

## Scoring Status

This section defines the scoring statuses used within this document. The scoring status indicates whether compliance with the given recommendation is discernable in an automated manner.

### *Scorable*

The platform's compliance with the given recommendation can be determined via automated means.

### *Not Scorable*

The platform's compliance with the given recommendation cannot be determined via automated means.

# Recommendations

# 1. Authentication mechanisms

This section provides guidance on the secure configuration of Sybase ASE authentication mechanisms and password policy recommendations.

Many large organizations will have defined a system-wide password policy in their Policies and Procedures documents dictating password complexity, expiry and lockout policy. The job of the database administrator is to ensure that the databases under their control enforce these policies. For organizations that do not have a rigid policy in place, the database administrator should enforce a strong policy whilst enabling business requirements.

## *1.1 Select an appropriate authentication mechanism (Level I, Not Scorable)*

**Description:**
Sybase ASE provides multiple means of authenticating users. These include Sybase proprietary authentication (username and password), Kerberos, LDAP user authentication (LDAPUA), secure LDAPUA and PAM user authentication (PAMUA). The Sybase LDAPUA implementation interoperates with LDAP v3 compliant servers such as Active Directory, iPlanet and OpenLDAP.

**Rationale:**
The most appropriate authentication mechanism depends on how Sybase is used within your organization. It is recommended that the System Security Officer consult the Sybase ASE 15.0 Administration Guide, Volume 1, Chapter 16 (External Authentication) for a discussion of the advantages and disadvantages of each.

**Remediation:**

1. Set appropriate authentication mechanism in accordance with your organization's security policy.

2. Ensure that the authentication mechanism is configured to not fallback to an alternative mechanism unless your organization's security policy explicitly permits this.

   For LDAPUA this is accomplished by connecting to the ASE server as a user with the `sso_role` and executing the following SQL statement:

   ```
   exec sp_configure "enable ldap user auth", 2
   ```

   For PAMUA this is accomplished by connecting to the ASE server as a user with the `sso_role` and executing the following SQL statement:

```
exec sp_configure "enable pam user auth", 2
```

**Audit:**

1. Perform the following to determine ASE's authentication mode:

```
exec sp_configure 'enable pam user auth'
exec sp_configure 'enable ldap user auth'
exec sp_configure 'use security services'
```

If the `Config Value` or `Run Value` returned for any of the above commands is non-zero, ASE will authenticate against that provider. If all `Config Value` and `Run Value` are zero, ASE will use Standard auithentication.

Additionally, Sybase supports per-login authentication providers. To determine if any logins are using a non-system-wide authentication provider perform the following for each login:

```
exec displaylogin '<login>'
```

The "Authenticate with:" label specifies the login's authentication provider.

**Additional References:**

1. Further information on authentication mechanisms within Sybase ASE 15 is contained within the technical whitepaper, Adaptive Server Enterprise Addresses Application Security Challenges, available at http://www.sybase.com/content/1030018/ASE_1252_Security_wp.pdf.

## *1.2 Review the default login (Level I, Scorable)*

**Description:**

When the login mode is set to Integrated or Mixed Mode authentication, domain usernames are mapped to database usernames via the `syslogins` table. If a domain user attempts to login to the database but has no corresponding `syslogins` entry, the user is logged in as the account specified by the `DefaultLogin` registry key.

This registry key is empty by default, indicating that only domain users with valid `syslogins` mappings may login. This setting should be reviewed to ensure that no default login has been set. If one has been set, its purpose should be fully established before it is modified in order to prevent disruption to applications and users that may be reliant upon

it. If similar functionality is required it can be accomplished by setting up a group within the Windows domain and creating a mapping within the `syslogins` table.

**Rationale:**
Assigning a value to the `DefaultLogin` registry key means that all users with valid Windows domain credentials have some level of access to the database. This goes against the security best practice principle of least privilege.

**Remediation:**

1. Set the value of the registry key `HKEY_LOCAL_MACHINE\SOFTWARE\SYBASE\Server\<ServerName>\DefaultLogin` to the empty string (where *<ServerName>* should be substituted for the name of the ASE instance).

**Audit:**

1. Ensure the value of the Registry key `HKEY_LOCAL_MACHINE\SOFTWARE\SYBASE\Server\<ServerName>\DefaultLogin` is set to the empty string (where *<ServerName>* should be substituted for the name of the ASE instance).

## *1.3 Store password hashes using SHA-256 only (Level I, Scorable)*

**Description:**
Sybase ASE 15.0.2 supports storing encrypted passwords using both SHA-256 hashes and the ASE proprietary algorithm or as SHA-256 hashes only. This setting is toggled via the `allow password downgrade` password policy option.

The default install setting for new ASE 15.0.2 installations is to store encrypted passwords as SHA-256 hashes only. ASE servers upgraded to 15.0.2 are set to also store encrypted passwords using the ASE proprietary algorithm.

Support for the ASE proprietary algorithm facilitates downgrades to older versions of Sybase ASE. If the System Administrator is certain that the ASE server will not be downgraded to an earlier version then encrypted passwords should be stored as SHA-256 hashes only.

Note that this configuration setting is not present ASE 15.0 or 15.0.1.

**Rationale:**
The SHA-256 algorithm is considered more secure than the ASE proprietary algorithm.

**Remediation:**

1. Connect to the database as a user with the `sso_role` and execute the following SQL statement to prevent the storage of encrypted passwords with the ASE algorithm:

```
exec sp_passwordpolicy 'set', 'allow password downgrade', 0
```

**Audit:**

1. Connect to the database as a user with the `sso_role` and execute the following SQL statement:

```
exec sp_passwordpolicy 'list', 'allow password downgrade'
```

2. For a new master database, the above statement should return the following to indicate that only SHA-256 is in use:

```
value    message
-------- ------------------------------------------------------
NULL     New master database.
```

3. For an upgraded database, the above statement should return the following to indicate that only SHA-256 is in use (where *<DateTime>* is the date and time on which the `allow password downgrade` option was set to 0):

```
value    message
-------- ------------------------------------------------------
0        Last Password downgrade was allowed on <DateTime>
```

## *1.4 Secure the sa account (Level I, Scorable)*

**Description:**
The System Administrator account, `sa`, is extremely powerful, having the `sa_role`, `sso_role`, `oper_role` and `Sybase_ts_role` by default. Furthermore, the password to the `sa` account is blank on install.

Sybase recommends using the `sa` account only for initial database configuration such as creating other users, devices and databases. It is then recommended that the `sa` account is locked.

The following steps represent best practice handling of the `sa` account:

- Set a strong password on the `sa` account; although the `sa` account should ultimately be locked, setting a strong password acts as a mitigating step while the database is being configured or should it be accidently re-enabled. This may have severe repercussions as the default password for the `sa` account is blank.
- Create separate user accounts and groups assigning the `sa_role`, `sso_role`, `oper_role` and `sybase_ts_role` roles as necessary, following the principle of least privilege.

- Remove the `sa_role`, `sso_role`, `oper_role` and the `sybase_ts_role` from the `sa` account.
- Lock the `sa` account

**Rationale:**
The first attack an intruder is likely to launch against Sybase ASE will be to test whether the `sa` account is enabled and whether it has a blank password. If both of these conditions are true, the attacker has no additional work to do to fully compromise the database. Furthermore, in many organizations auditing requirements mandate the user of non-default accounts so that operations can be accurately tracked.

**Remediation:**

1. Connect to the ASE server as a user with the `sa_role` and execute the following SQL statement to set a strong password on the `sa` account (where *<StrongPassword>* should be substituted for a suitable password):

```
exec sp_password NULL, '<StrongPassword>'
```

2. Ensure the above statement returns `Password correctly set.`

3. Create separate user accounts and groups assigning the `sa_role`, `sso_role`, `oper_role` and `sybase_ts_role` roles as necessary, following the principle of least privilege.

4. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statements, ensuring they complete successfully, to strip the `sa` account of the `sso_role`, `oper_role` and `sybase_ts_role` roles.

   Note that it is essential that other accounts with at least the `sa_role` and the `sso_role` have been created prior to carrying out this and the proceeding step.

```
revoke role sso_role from 'sa'
revoke role oper_role from 'sa'
revoke role sybase_ts_role from 'sa'
```

5. Connect to the ASE Server as user with the `sa_role` and executing the following statement, ensuring it completes successfully, to strip the `sa` account of the `sa_role`.

```
revoke role sa_role from 'sa'
```

6. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement to lock the `sa` account:

```
exec sp_locklogin sa, 'lock'
```

7. Ensure the above statement returns `Account locked`.

**Audit:**

1. To verify that the `sa` account has a non-NULL password, you must attempt to connect to Sybase ASE with a NULL password. The `syslogins.password` field will not be NULL even if a NULL password is present. This validation step must be performed prior to locking the `sa` account. If the account's lock status is unknown, it is not feasible to determine if its password is NULL.

2. As an optional step to ensure that a strong password is set, run a password cracking tool on the encrypted password returned from the query.

3. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement to verify that the `sa` account does not have privileged roles:

```
exec sp_displaylogin sa
```

4. Ensure that the text returned under `Configured Authorization` does not contain any roles.

5. Ensure that the text returned by `Locked` reads `YES` and that the text returned by `Reason` reads `Account locked by ASE by manually executing sp_locklogin`.

   Note: `Reason` is only available in ASE 15.0.2 and greater.

## *1.5 Remove unused accounts and change default passwords (Level I, Scorable)*

**Description:**
Many Sybase components that interact with ASE create user accounts with weak passwords such as "Sybase", "SQL" or the username itself.

It is recommended that default accounts are given passwords that conform to a strong password policy. Furthermore, accounts that are not in use should be removed. Below is a list of common accounts to inspect:

- probe

- sybmail
- jstask
- mon_user

**Rationale:**
Default passwords present an easy means of compromising a database, even for unskilled attackers. Even if the targeted user account does not have access to powerful roles or sensitive data, the attacker need only find a privilege escalation vulnerability to execute a full compromise.

**Remediation:**

1. Connect to the ASE server as a user that has `select` permission on `master.dbo.syslogins` (such as a user with the `sso_role`) and execute the following SQL statement to retrieve a list of database usernames:

```
use master
select name from syslogins
```

2. Set strong passwords on these accounts via the `sp_password` stored procedure and ensure all client components that make use of the account are updated to use the new password.

**Audit:**

1. Connect to the ASE server as a user that has `select` permission on `master.dbo.syslogins` (such as a user with the `sso_role`) and execute the following SQL statement to retrieve a list of database usernames:

```
use master
select name from syslogins
```

2. Ensure the above list includes only required accounts.

**Additional References:**

1. A list of default passwords for Sybase components is provided in the "Guide to Sybase Security" whitepaper by Nilesh Burghate, NII Ltd. It is available at http://www.niiconsulting.com/innovation/Sybase.pdf.

## *1.6 Enforce password complexity (Level II, Scorable)*

**Description:**
Sybase ASE 15.0 and 15.0.1 supports enforcing password complexity via:

- Setting the login mode to Integrated Mode so that password policy is enforced by the Windows domain.

- A configuration parameter to enforce server-wide, per user account and per role minimum password length (set to 0 by default)
- A configuration parameter to enforce at least one digit in a password (disabled by default)

Sybase ASE 15.0.2 supports the above settings as well as more granular password complexity via:

- A setting to enforce that a login name cannot be a substring of the password.
- A setting to enforce the minimum number of special characters for the password.
- A setting to enforce the minimum number of alphabetic characters for the password
- A setting to enforce the minimum number of upper-case letters for the password.
- A setting to enforce the minimum number of lower-case letters for the password.
- A setting to enforce that the password must be reset is the first time a login is used.
- A setting to enforce the minimum number of digits for the password.

In addition, Sybase ASE 15.0.2 supports the creation of a stored procedure to enforce custom password complexity requirements.

It is recommended that strong password complexity is enforced in accordance with your organization's policy. It may not be possible to enforce a sufficient policy on ASE 15.0 and ASE 15.0.1; if this is the case the System Security Officer should consider one of the following solutions:

- Upgrade systems to ASE 15.0.2 in order to make use of the more extensive password complexity options.
- Enable Integrated Mode to rely on the Windows domain password policy.
- Accept the risk associated with the policy conflict and regularly audit password strength using a password cracking tool.

**Rationale:**
Arguably the most common cause of database compromise is weak passwords. Setting password complexity is essential step to ensuring the security and integrity of the data within the database.

**Remediation for ASE 15.0 and 15.0.1:**
Perform the following to enable password complexity requirements while operating in Standard login mode:

1. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement in order to set a system-wide minimum password length according to your organization's password (substitute 8 for an acceptable value):

```
exec sp_configure 'minimum password length', 8
```

2. Set a custom minimum password length for specific users and roles as required. This should not be less than the system-wide length. This can be accomplished via the `sp_modifylogin` stored procedure.

3. Execute the following statement to enforce at least one digit in passwords:

```
exec sp_configure 'check password for digit', 1
```

**Audit:**
Perform the following to audit password complexity requirements while operating in Standard login mode:

1. Connect to the ASE server (the `sso_role` is not required) and execute the following SQL statement to confirm a system-wide minimum password length is enforced:

```
exec sp_configure 'minimum password length'
```

2. The `Config Value` and `Run Value` should represent the desired minimum password length:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 minimum password length                8           0            8            8 bytes               dynamic
```

3. Verify that important user accounts have a custom minimum password length as required.

4. Execute the following statement to verify that password require at least one digit:

```
exec sp_configure 'check password for digit'
```

5. The `Config Value` and `Run Value` should be `1`.

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 check password for digit               0           0            1            1 switch              dynamic
```

**Remediation for ASE 15.0.2:**
Perform the following to audit password complexity requirements while operating in Standard login mode:

1. Connect to the ASE server as a user with the `sso_role` and execute the `sp_passwordpolicy` stored procedure. The following options can be used to configure a password policy in accordance with your organization's requirements:

| | |
|---|---|
| `disallow simple passwords` | A value of 1 turns this option on, and a value of 0 turns it off. |
| `min digits in password` | Indicates the minimum number of digits to be allowed in a password. |
| `min alpha in password` | Indicates the minimum number of alphabetic characters in a password. |
| `min special char in password` | Indicates the minimum number of special characters allowed in a password. |
| `min upper char in password` | Indicates the minimum number of upper case characters allowed in a password. |
| `min lower char in password` | Indicates the minimum number of lower case characters allowed in a password. |
| `systemwide password expiration` | Indicates the system wide password expiration in days. |
| `password exp warn interval` | Indicates the password expiration warning interval in days. |
| `minimum password length` | Sets the minimum length of the password. |
| `maximum failed logins` | Sets the maximum number of failed logins allowed in a session before the account is locked. |
| `expire login` | Specifies that a login status changes to expired status when the SSO creates a login or a user reset their login. The user will be required to change their password on their first login. |

2. Implement required custom password checks (e.g. to verify the password is not based on a word associated with your organization) via creating the following stored procedures in the `master` database:

```
sp_extrapwdchecks caller_password, new_password, login_name
sp_cleanpwdchecks, login_name
```

Further information on creating these stored procedures is available in the New Features Guide for Adaptive Server Enterprise 15.0.2.

**Audit:**

1. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement to retrieve the password policy settings in effect:

```
exec sp_passwordpolicy 'list'
```

2. Verify that the password policy returned is in accordance with your organization's requirements.

## *1.7 Set lockout thresholds (Level II, Scorable)*

**Description:**
Sybase ASE supports setting lockout thresholds that define the number of incorrect consecutive login attempts that will result in the account or role being locked. These can be specified on a global basis (i.e. applicable to all user accounts), on a per user basis and on a per role basis with individual settings overriding server-wide settings.

The default lockout threshold in Sybase ASE allows unlimited incorrect login attempts. At a minimum, a global lockout threshold should be set in accordance with your organization's password policy. It is recommended that user accounts that have powerful roles such as `sa_role` or `sso_role` should have a stricter threshold set.

**Rationale:**
Allowing an attacker unlimited attempts to login to an account permits a brute force attack to proceed unhindered, potentially leading to compromise of the database.

**Remediation:**
1. Connect to the ASE server with a user that has the `sso_role` and execute the following SQL statement (note "5" should be substituted for the lockout threshold required within your organization):

```
exec sp_configure 'maximum failed logins', 5
```

**Audit:**
1. Connect to the ASE server (the `sso_role` is not required) and execute the following SQL statement:

```
exec sp_configure 'maximum failed logins'
```

2. The `Config Value` and `Run Value` should represent the desired lockout threshold.

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
 ------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 maximum failed logins                    0           0            5            5 number               dynamic
```

## *1.8 Set a system-wide password expiration (Level II, Scorable)*

**Description:**
Sybase ASE supports expiring passwords after a set interval. The interval can be set on a global, per user or per role basis. Password expiration is disabled by default.

It is recommended that a system-wide password expiration is set according to your organization's requirements.

**Rationale:**
Password expiration potentially mitigates the damage from a compromised account. It also assists in identifying accounts that are no longer in use.

**Remediation:**

1. Connect to the ASE server with a user that has the `sso_role` and execute the following SQL statement to set the system-wide password expiration (substitute 90 for a suitable password expiration value based on your organization's requirements):

```
exec sp_configure 'systemwide password expiration', 90
```

**Audit:**

1. Connect to the ASE server (the `sso_role` is not required) and execute the following SQL statement to retrieve the system-wide password expiration:

```
exec sp_configure 'systemwide password expiration'
```

2. The `Config Value` and `Run Value` should represent the desired lockout threshold:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 systemwide password expiration           0           0           90           90 days                 dynamic
```

## *1.9 Set passwords on important roles (Level II, Scorable)*

**Description:**
Sybase ASE supports setting role passwords to ensure that all users have to enter a password before using a protected role. Powerful roles such `sa_role` and `sso_role` should be password protected to provide as an additional layer of security.

**Rationale:**
Password protecting powerful roles means that even if a user is granted that role (which might have been granted through error or indirectly via the `WITH GRANT` option) they must know the password to use it.

**Remediation:**

1. Connect to the ASE server with a user that has the `sa_role` and execute the following SQL statement (where <Role> should be substituted for the role that is to be password protected and <Password> for the new password):

```
set role "<Role>" with password "<Password>"
```

**Audit:**

1. Connect to the ASE server with a user that has the `select` permission on the `master.dbo.syssrvroles` table (e.g. a user with the `sa_role`) and execute the following SQL statement:

```
use master
select name from syssrvroles where password = NULL
```

2. The role names returned do not currently have a password set. Ensure that this is acceptable.

## *1.10 Use login triggers to validate users' IP addresses (Level II, Not Scorable)*

**Description:**
Sybase ASE supports login triggers; these execute a specified stored procedure every time a user logs in. Login triggers can be used to carry out additional verification steps such as checking the IP address that the user is logging in from is as expected.

Note: Global login triggers are available on ASE 15.0.2 and greater.

**Rationale:**
Login triggers can provide an additional layer of security through verification of criterion such as IP address. Note that the IP address may be subject to spoofing or may indicate a compromised client and as such should not be exclusively relied upon.

**Remediation:**

1. Connect to the ASE server with a user that has the `sso_role` and execute the following SQL statement where *<Login_Name>* should be substituted for the username on which the login trigger will fire and *<Sproc_Name>* for the specific stored procedure. If *<Login_Name>* is set to `NULL`, a global login trigger is registered (i.e. for all users). Global login triggers can also be set via the `sp_logintrigger` stored procedure.

```
exec sp_modifylogin <Login_Name>, "login script", <Sproc_Name>
```

Note that the stored procedure registered as a login trigger must be available in the user's default database since Sybase ASE searches the `sysobjects` table in the user's default database in order to find the login trigger object.

**Audit:**

1. Connect to the ASE server with a user that has the `sso_role` and execute the following SQL statement where <Login_Name> should be substituted for the username for which the login trigger status is being determined:

```
exec sp_displaylogin <Login_Name>
```

2. Verify that the "Auto Login Script" value returned contains the expected stored procedure name.

3. Determine the presence of a global login trigger via connecting to the ASE Server with a user that has the `sso_role` and executing the following SQL statement:

```
exec sp_logintrigger
```

4. Verify that the stored procedure name returned is as expected and its status is set to "Enabled".

## *1.11 Conceal Sensitive Input to isql (Level I, Not Scorable)*

**Description:**

The Open Server 15.0 and SDK 15.0 client components for Sybase ASE, ESD #13 and above, support concealment of input during isql sessions via the **`--conceal`** command line option.

**Rationale:**
The `--conceal` option is useful when entering sensitive information, such as passwords in environments where echoed console input may be visible to multiple parties.

**Remediation:**

1. Specify the `--conceal` command line option as follows, where <Wildcard> is the character string that triggers isql to prompt for concealed input (note that this default to the string ':?' if no wildcard is supplied):

```
isql --conceal '<Wildcard>'
Complete this example by including a command (like sp_password) and
demonstrating how passwords are "concealed".
```

**Audit:**

1. Verify the isql sessions make use of the `--conceal` command line argument.

**Additional References:**

1. Further information on the –conceal option may be found in the New Features Open Server 15.0 and SDK 15.0 for Microsoft Windows, Linux, and UNIX document

available at
http://infocenter.sybase.com/help/topic/com.sybase.dc20155_1500/pdf/newfesd.pdf.

# 2. Network Security Mechanisms

This section provides guidance on the recommended configuration of the network options within Sybase ASE. The goal of these settings is to configure the server such that it enforces a sufficient level of encryption, both when operating as a server and as a client (when connecting to a remote server) in order to prevent both active network-based attacks such as man-in-the-middle and passive attacks such as packet sniffing to capture passwords.

## *2.1 Enable Secure Socket Layer (SSL) Encryption (Level II, Scorable)*

**Description:**
Sybase ASE supports SSL encryption as a means of ensuring confidentiality between clients and servers. SSL is a widely accepted standard for securing the transmission of sensitive information, such as credit card numbers, stock trades, and banking transactions over the Internet. It relies on public-key cryptography and allows the client and server to negotiate a mutually acceptable cipher.

Sybase ASE 15.0.2 also supports the NIST-approved AES algorithm and new options for setting cipher suite preference via the `sp_ssladmin` stored procedure.

SSL encryption of connections is disabled by default. When it is enabled, a client can potentially negotiate a cryptographically weak cipher suite. It is recommended that SSL support is enabled and that the cipher suite preference is set to `strong` (or `FIPS` if your organization mandates FIPS-compliance).

**Rationale:**
SSL encryption prevents passive sniffing attacking from capturing sensitive data that may be transmitted between client applications and the server such as credit cards and SSNs. A strong cipher suite is required to prevent cryptographic attacks on clients that force the server to use weak algorithms.

**Remediation:**
There are several steps involved to enable SSL. The information below is provided as an outline only; it is recommended that the System Security Officer consult the *Sybase ASE 15.0 Administration Guide, Volume 1, Chapter 19* for detailed advice.

1. Generate a certificate for the server.

2. Create a trusted roots file.

3. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement to enable SSL:

```
exec sp_configure "enable ssl", 1
```

4. Add the SSL filter to the interfaces file.

5. Use `sp_ssladmin` stored procedure to add a certificate to the certificates file. See "Administering certificates".

6. Execute the following SQL statement to enforce strong cipher suites (note 'strong' should be substituted for 'FIPS' if your organization mandates the use of FIPS-compliant algorithms):

```
exec sp_ssladmin setcipher, 'strong'
```

**Audit:**

1. Connect to the ASE server (the `sso_role` is not required) and execute the following SQL statement to verify that SSL is enabled:

```
exec sp_configure "enable ssl"
```

2. The `Config Value` column will be set to `1` if SSL is enabled.

3. Execute the following SQL statement to retrieve the configured cipher suite preference:

```
exec sp_ssladmin lsciphers
```

4. Ensure that the ciphers returned are acceptable. If no ciphers are returned then cipher preference has not been set and Sybase will use the default ciphers.

## *2.2 Enable message integrity (Level I, Scorable)*

**Description:**
Sybase ASE supports a means of signaling to the underlying security mechanism that message integrity is required via the `msg integrity reqd` configuration parameter.

The setting is disabled by default. It is recommended the message integrity is enabled. Note that enabling the `use security services` configuration parameter is a prerequisite for enabling message integrity.

**Rationale:**
Enabling message integrity prevents an attacker positioned between the client and the server from intercepting and modifying messages.

**Remediation:**

1. Connect to the database as a user with the `sso_role` and execute the following SQL statement to enable message integrity.

```
exec sp_configure "msg integrity reqd", 1
```

**Audit:**

1. Connect to the database (the `sso_role` is not required) and execute the following SQL statement:

```
exec sp_configure "msg integrity reqd"
```

2. The `Config Value` and `Run Value` returned should be 1:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 msg integrity reqd                      0           0            1            1 switch               dynamic
```

## *2.3 Enable message confidentiality (Level I, Scorable)*

**Description:**
Sybase ASE supports a means of signaling to the underlying security mechanism that message confidentiality via encryption is required. This is accomplished through the `msg confidentiality reqd` configuration parameter.

The setting is disabled by default. It is recommended the message confidentiality is enabled. Note that enabling the `use security services` configuration parameter is a prerequisite for enabling message confidentiality.

**Rationale:**
Enabling message confidentiality prevents an attacker positioned between a client and the servers from being able to capture sensitive data.

**Remediation:**

1. Connect to the database as a user with the `sso_role` and execute the following SQL statement to enable message confidentiality.

```
exec sp_configure "msg confidentiality reqd", 1
```

**Audit:**

1. Connect to the database (the `sso_role` is not required) and execute the following SQL statement:

```
exec sp_configure "msg confidentiality reqd"
```

2. The `Config Value` and `Run Value` returned should be 1:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 msg confidentiality reqd                0           0            1            1 switch               dynamic
```

## *2.4 Enable network password encryption (Level I, Scorable)*

**Description:**
Sybase ASE 15.0.2 supports the use of asymmetric encryption to securely transmit passwords from the client to the server using the RSA public key encryption algorithm. This setting is enabled via the `net password encryption reqd` configuration parameter. This feature does not depend on PKI, Kerberos, nor SSL.

There are three possible settings for the value of the `net password encryption reqd` configuration parameter:

- 0 – This setting allows the client to choose the encryption types, including no encryption. This is the default settings.
- 1 – This setting causes the server to permit either the older proprietary ASE encryption or the RSA algorithm only.
- 2 – This setting causes the server to permit only the RSA algorithm.

If all client applications within your organization support the RSA algorithm (i.e. they use client libraries accompanying ASE 15.0.2 or are RSA algorithm aware) then it is recommended that setting 2 is enabled, otherwise it is recommended that setting 1 is enabled.

Note that this setting is not supported by ASE 15.0 or 15.0.1.

**Rationale:**
Enabling network password encryption prevents an attacker positioned between the client and the server from sniffing the password during the login process. The RSA algorithm is preferred over the proprietary ASE algorithm since RSA is a widely accepted and analyzed algorithm.

**Remediation:**
1. Connect to the database as a user with the `sso_role` and execute the following SQL statement to set the network password encryption to 2:

```
exec sp_configure "net password encryption reqd", 2
```

**Audit:**

1. Connect to the database (the `sso_role` is not required) and execute the following SQL statement:

```
exec sp_configure "net password encryption reqd"
```

2. The `Config Value` and `Run Value` returned should be 2:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 net password encryption reqd             0           0            2            2 switch               dynamic
```

## 2.5 Remote Server Security Settings

Sybase ASE supports Remote Procedure Calls (RPCs) allowing users on a local Adaptive Server to execute stored procedures on a remote server. If an attacker is able to position themselves between two servers either directly or via a Layer 2 attack from elsewhere on the network, such as ARP spoofing, the default RPC settings allow for disclosure of sensitive information such as account passwords or even direct compromise of the database.

The following settings should be applied if RPC is in use.

### *2.5.1 Enable password encryption (Level I, Scorable)*

**Description:**
When a local Sybase ASE server connects to a remote server, the user account password is sent across the network encrypted or in plain text dependant on the `net password encryption` setting for the server.

- On Sybase ASE 15.0, `net password encryption` is set to false by default.
- On Sybase ASE 15.0.1, `net password encryption` is set to false by default.
- On Sybase ASE 15.0.2, `net password encryption` is set to false by default (for any new server added using `sp_addserver` and for `sysservers` entries with an ASEnterprise class value during upgrade to this release).

The `net password encryption` should be set to true for each remote server.

**Rationale:**
An attacker that is able to sniff the traffic between two servers would be able to capture passwords that are sent in plain text.

**Remediation:**

1. Connect to the database as a user with the `sso_role` and execute the following SQL statement (where *<ServerName>* represents the name of the remote server for which password encryption will be enabled):

```
exec sp_serveroption <ServerName>, 'net password encryption', true
```

**Audit:**

1. Connect to the database (the `sso_role` is not required) and execute the following SQL statement (where *<ServerName>* represents the name of the remote server for which password encryption is being audited):

```
exec sp_helpserver <ServerName>
```

2. The status returned should contain the string `net password encryption`:

```
name         network_name   class        status                                                                    id cost
------------ -------------- ------------ ------------------------------------------------------------------------- -- ----
 <ServerName> <ServerName>   ASEnterprise no timeouts, net password encryption, writable , rpc security model B  5 1000
```

## *2.5.2 Consider disabling remote access (Level II, Scorable)*

**Description:**
Sybase ASE allows server-to-server RPC to be disabled via the `allow remote access` configuration parameter. By default server-to-server RPC is enabled since it is required for communication with the Backup Server; disabling server-to-server RPC will make it impossible to back up a database.

The Sybase System Administrator Guide for ASE 15.0, Volume 1 Chapter 5 claims:

*Since other system administration actions are required to enable remote servers other than Backup Server to execute RPCs, leaving this option set to 1 does not constitute a security risk.*

Nonetheless, if communication with remote servers including the Backup Server is **not** required then this configuration parameter can be set to 0 to disable server-to-server RPC.

**Rationale:**
Disabling remote access will reduce the remote attack surface of system.

**Remediation:**

1. Connect to the database as a user with the `sso_role` and execute the following SQL statement to disable server-to-server RPC:

```

```

```
exec sp_configure 'allow remote access', 0
```

**Audit:**

1. Connect to the database (the `sso_role` is not required) and execute the following SQL statement:

```
exec sp_configure "allow remote access"
```

2. The `Config Value` and `Run Value` returned should be 0:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 allow remote access                     1           0            0            0 switch               dynamic
```

# 3. Database resource permissions

## *3.1 General Resources*

### *3.1.1 Set an appropriate default database for all users (Level I, Scorable)*

**Description:**
It is recommended that no users except those who have the `sa_role` or `sso_role` are assigned a default database of `master`, since this database stores all system tables.

**Rationale:**
As a general best practice recommendation, all standard users should be associated with a specific "home" database other than `master`.

**Remediation:**

1. Connect to the ASE server as a user that has `select` permission on `syslogins` (e.g a user with the `sa_role)` and execute the following SQL statement to retrieve the list of users that currently have a default database of `master`:

```
use master
select name, dbname from syslogins where dbname = 'master'
```

2. For each user that has a default database of `master`, that does not have the `sa_role` and/or the `sso_role` (role membership can be determined via the `sp_displaylogin` stored procedure), execute the following SQL statement to modify their default database. *<Login>* should be substituted for the appropriate username and *<Database>* for the new default database to be set.

```
```

```
exec sp_modifylogin <Login>, defdb, <Database>
```

**Audit:**

1. Connect to the ASE server as a user that has `select` permission on `syslogins` (e.g a user with the `sa_role)` and execute the following SQL statement to retrieve the list of users that currently have a default database of `master`:

```
use master
select name, dbname from syslogins where dbname = 'master'
```

2. Ensure that the only users returned have the `sa_role` and/or the `sso_role`.

### *3.1.2 Restrict use of set proxy (Level I, Not Scorable)*

**Description:**
Sybase ASE supports proxy authorization, allowing Security Officers the ability to grant selected logins the ability to assume the security context of another user via the `set proxy` grant.

**Rationale:**
The `set proxy` grant potentially allows a user to impersonate any other login unless restricted via the `restrict role` parameter.

**Remediation:**

1. When using the `set proxy` command, always use the `restrict role` parameter.

**Audit:**

1. Connect to the database and execute the following SQL statement:

```
sp_helprotect @permission_name = 'Set Proxy'
```

## *3.2 Database Users*

### *3.2.1 Review use of the guest user in databases (Level II, Scorable)*

**Description:**
Adding a guest entry to the `sysusers` table of any database effectively permits any database user to use the database with the permissions of the `guest` user (which by default inherits the permissions of the `public` role).

Rather than using the `guest` user it is recommended that roles be set up within Sybase ASE to facilitate multiuser access to databases.

**Rationale:**

Adding a guest entry to a database goes against the security best practice principle of least privilege and makes it harder to audit operations.

**Remediation:**

1. Identify the databases that contain a guest user.

2. Identify the users that access objects in these databases.

3. Either grant each user specific access to each database as required or create appropriate roles and grant each role specific access to each database.

**Audit:**

1. Connect to the database as a user with the `sa_role` and execute the following SQL statement to determine which databases contain a guest entry in their `sysusers` table:

```
exec sp_helpuser guest
```

## *3.3 Data Access*

### *3.3.1 Avoid use of `grant all` (Level I, Not Scorable)*

**Description:**
When granting or revoking privileges to a database object, Sybase ASE allows the syntax `grant all` to signify that all privileges applicable to the specified object should be granted or revoked. It is recommended that use of `grant all` is avoided where possible.

**Rationale:**
Security best practice advocates the principle of least privilege, i.e. only the privileges that are absolutely necessary should be granted to a user. In situations where all privileges are not required, use of `grant all` violates this principle.

**Remediation:**

1. Use specific grant statements to assign the required privileges.

**Audit:**

1. Regularly review assigned privileges via the `sp_helprotect` stored procedure.

### *3.3.2 Limit access via procedures, views and triggers (Level II, Not Scorable)*

**Description:**
Sybase ASE supports views and stored procedures as security mechanisms, allowing a user (role or group) to be granted permission on a view or on a stored procedure even if they have no permissions on objects the view or procedure references.

**Rationale:**
By defining different views and stored procedures and selectively granting permissions on them, a user (or any combination of users) can be restricted to different subsets of data allowing for a granular implementation of security requirements.

**Remediation:**
1. Identify the subsets of data that should be accessible to particular users. Implement views and triggers as described in Sybase ASE System Administration Guide, Volume 1, chapter 17.

**Audit:**
1. Periodically review the user (role and group) requirements for access to data updating the views and triggers appropriately.

## *3.4 Revoke default permissions for the public role (Level I, Scorable)*

**Description:**
By default, the `public` role has select permission on many system tables, including the `syslogins` table in the `master` database (though not on the `password` column).

Since all database users have the `public` role it is recommended that these permissions are revoked from all databases.

This setting should be thoroughly tested on non-production servers before it is applied since additional table privileges may need to be granted to specific users or groups once `public` access is revoked.

**Rationale:**
Low privileged database users can glean useful information from system tables such as account names and lockout status.

**Remediation:**
1. Connect to the ASE server as a user with the `sa_role` and execute the following SQL statement for each database listed in `sysdatabases` (where <Database> should be substituted for the appropriate database name). For the complete list of tables that this command affects, see the description of the `revoke` command in the Sybase ASE Reference Manual: Commands.

   Note that the tables affected differ depending on whether the database is the `master` database or not.

```
use <Database>
revoke default permissions on system tables
```

**Audit:**

1. Connect to the ASE server as a user with the `sa_role` and execute the following SQL statement for each database listed in `sysdatabases`.

```
use <Database>
exec sp_helprotect
```

2. Ensure that `public` does not have `select` permission on any system tables.

## *3.5 Ensure updates to system tables are not permitted (Level I, Scorable)*

**Description:**
Sybase ASE can protect system tables from direct or accidental alteration through SQL queries via the `allow updates to system tables` configuration parameter.

This setting is enabled by default. It is recommended that this setting is re-enabled if it has been disabled.

**Rationale:**
An attacker with sufficient privilege can re-enable direct updates to system tables, but this configuration setting should protect against accidental alterations and will aid the audit trail.

**Remediation:**

1. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement:

```
exec sp_configure 'allow updates to system tables', 0
```

**Audit:**

1. Connect to the ASE server (the sa_role is not required) and execute the following SQL statement:

```
exec sp_configure 'allow updates to system tables'
```

2. The `Config Value` and `Run Value` returned should be 0:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 allow updates to system tables           0           0            0            0 switch               dynamic
```

### *3.5.1 Protect database object text in syscomments (Level I, Scorable)*

**Description:**

The `syscomments` table contains the source code for business logic implementation such as stored procedures. It also contains the text of views, triggers, default table constraints, and procedures. By default the `public` role has `select` permission on this system table.

Sybase ASE supports a configuration parameter, `select on syscomments.txt`, that restricts `select` permission to the object owner and users with the `sa_role`. It is recommended that this configuration is enabled.

**Rationale:**
`select` permission should be restricted to the object owner and system administrators only since stored procedures, triggers and views often contain sensitive information. Furthermore, source code access is likely to facilitate the discovery of logic flaws that may result in privilege escalation or information disclosure.

**Remediation:**

1. Connect to the database as a user with the `sso_role` and execute the following SQL statement:

```
exec sp_configure 'select on syscomments.text', 0
```

**Audit:**

1. Connect to the database (the `sso_role` is not required) and execute the following SQL statement:

```
exec sp_configure 'select on syscomments.text'
```

2. The `Config Value` and `Run Value` returned should be 0:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 select on syscomments.text               1           0            0            0 switch               dynamic
```

## 3.6 Encryption settings

Sybase ASE 15.0.1 and above supports the encryption of data at rest at column-level granularity without the need to modify client applications to support it. Data is encrypted with the NIST approved AES algorithm.

This section contains best practice recommendations for configuring Sybase ASE encryption settings.

### *3.6.1 Ensure a strong system encryption password is set (Level I, Scorable)*

**Description:**

It is the responsibility of the System Security Officer to set a strong system encryption password. This password is used to generate a 128-bit key-encrypting key, which in turn is used to encrypt column encryption keys (created by users with the `create encryption key` privilege).

**Rationale:**
Setting a weak system encryption password facilitates the decryption of column encryption keys and ultimately the data itself.

**Remediation:**

1. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement to set a system encryption password (where *<Password>* should be substituted for the strong system encryption password). Note that support for encrypted columns must be enabled before the system encryption password can be set.

```
exec sp_encryption system_encr_passwd, '<Password>'
```

**Audit:**

1. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement to determine if a system encryption password is set.

```
exec sp encryption helpkey, system encr passwd
```

### *3.6.2 Store encryption keys in a separate database (Level II, Scorable)*

**Description:**
Sybase ASE allows columns to be encrypted with keys that reside in the same database or in different databases. Encryption keys should be stored in a separate database from the data that they are used to encrypt.

**Rationale:**
In the event of the theft of a database dump, the attacker must have access to dumps of the encryption key database and the database holding the encrypted data rather than a single database that holds both the keys and the encrypted data.

**Remediation:**

1. Connect to the ASE server as a user with the `sso_role` or the `keycustodian_role` and execute the following SQL statement to create an encryption key in a specified database (where *<Database>* should be substituted for the database that is to hold the encryption key, *<Owner>* for key owner and *<KeyName>* for the key name). Note that the following statement is provided as an example only; the Sybase ASE 15.0.2 Reference Manual contains the full syntax for the `create encryption key` command.

```
create encryption key <Database>.<Owner>.<KeyName>
```

**Audit:**

1. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statements to verify that the database holding encrypted data does not contain encryption keys (where *<Database>* should be substituted for the database holding encrypted data).

```
use <Database>
exec sp_encryption helpkey
```

2. Verify that the text `There are no encryption keys (key copies) like '%' in '<Database>'` is returned, indicating that the database holds no encryption keys.

### *3.6.3 Password protect encryption keys (Level II, Scorable)*

**Description:**
Sybase ASE 15.0.2 supports per encryption key passwords that can be used to restrict access to encrypted data. This can be used to limit DBO and system administrator access to data; a user must have knowledge of the encryption key password as well as the decrypt permission on the column in order to decrypt the data.

**Rationale:**
Depending on your organization's security policy it may be a requirement to restrict data access to a small subset of users that excludes system administrators; encryption key passwords provide a means of accomplishing this within Sybase ASE.

**Remediation:**

1. Connect to the ASE server as a user with either the `sso_role` or the `keycustodian_role` and execute the following SQL statement to create an encryption key with a password (where *<KeyName>* should be substituted for the chosen key name and *<Password>* for a strong password). Note that the following statement is provided as an example only; the Sybase ASE 15.0.2 Reference Manual contains the full syntax for the `create encryption key` command.

```
create encryption key <KeyName> with passwd '<Password>'
```

**Audit:**

1. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement to verify that passwords are set on each encryption key within the

specified database (where *<Database>* should be substituted for the database holding encryption keys).

```
use <database>
go

exec sp_encryption helpkey
```

2. Verify that none of the keys reported by this command contain “System Encr Passwd” in the column “Type of Password”.

# 4. Auditing, Logging and Reporting Mechanisms

The auditing of security related events and the periodic analysis or alert-based monitoring of audit logs are essential operations in order to detect ongoing and past database attacks, both successful and unsuccessful. Though signature and anomaly-based intrusion detection systems can be useful, they are often unable to detect customized attacks. Consequently the most useful audit log is often likely to be on the database itself, provided at least basic auditing has been configured.

This section provides guidance on the recommended configuration of the auditing options within Sybase ASE.

## *4.1 Ensure sufficient space for logs (Level II,Scorable)*

**Description:**
Logging plays a critical part in ensuring database consistency and integrity. Disk space should be monitored to ensure that Sybase ASE has sufficient space to store logs.

**Rationale:**
A denial of service condition will occur if Sybase ASE runs out of space for logging. If the database serves as a backend for a web application, it may be possible for a web application user to trigger such a condition by repeatedly requesting dynamic resources.

**Remediation:**

1. It is recommended that a separate partition is used for storing logs and that thresholds are set up so that the `sp_thresholdaction` stored procedure is executed when the threshold is crossed.

**Audit:**

1. Ensure a separate partition is used for storing logs and that there are thresholds in place.

## *4.2 Enabling resource limits (Level I, Scorable)*

**Description:**
Sybase ASE provides a means of resource limiting via the `allow resource limits` configuration parameter. This functionality is disabled by default. When it is enabled, the server applies limits to user sessions.

It is recommended that this setting is enabled to mitigate against denial of service and data mining attacks. This setting should be thoroughly tested on non-production servers to ensure that it does not interfere with normal application behavior.

**Rationale:**
Resource limiting may be a useful defense against potential attacks aimed at denial of service or data mining attacks (e.g. through SQL Injection).

**Remediation:**

1. Connect to the ASE server as a user with the `sa_role` and execute the following SQL statement to enable resource limits. Note that the ASE Server must be restarted for this configuration to take effect.

```
exec sp_configure 'allow resource limits', 1
```

**Audit:**

1. Connect to the ASE server (the `sa_role` is not required) and execute the following SQL statement:

```
exec sp_configure 'allow resource limits'
```

2. The `Config Value` and `Run Value` returned should be 1:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 Allow resource limits                    0          #4            1            1 switch               static
```

## *4.3 Enable auditing (Level I, Scorable)*

**Description:**
Auditing is disabled by default in Sybase ASE. It should be enabled and at a minimum and configured to audit the following events:

- All commands which require the `sa_role`
- All errors
- All logins to the database

In addition, audit settings should also be configured to detect significant departures from typical business use such as execution of unused stored procedures as well as the creation and modification of database objects. This may mean auditing `GRANT`, `DROP` and `CREATE` actions as well.

Auditing settings should be thoroughly tested on non-production systems to ensure they do not impact performance on database with heavy usage.

**Rationale:**
Auditing of security-related events is essential to ensure the security of the database and the integrity of the data held within it.

**Remediation:**

1. Install the auditing functionality. This is a multistage process involving the following steps:

   - Creation of the auditing "devices".
   - Creation of the auditing database.
   - Running the `installsecurity (instsecu` on Windows) script to populate the database tables.
   - Restarting the database.

   For detailed information, see
   - the Sybase ASE Configuration Guide for your platform (Windows or UNIX), Chapter on "Adding Optional Functionality to Adaptive Server",
   - the Sybase ASE System Administration Guide chapter on "Auditing".

2. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement to enable auditing of security-related events, errors and login attempts:

```
exec sp_configure 'auditing', 1

/* Enable auditing of all security-related events for all users */
exec sp_audit 'security', 'all', 'all', 'on'

/* Enable auditing of all errors for all users */
exec sp_audit 'errors', 'all', 'all', 'on'

/* Enable auditing of all logins for all users */
exec sp_audit 'login', 'all', 'all', 'on'
```

3. Configure additional auditing options as required.

**Audit:**

1. Connect to the ASE server (`sso_role` is not required) and execute the following SQL statement:

```
exec sp_configure 'auditing'
```

2. The `Config Value` and `Run Value` returned should be 1 indicating that auditing is enabled:

```
Parameter Name                Default     Memory Used Config Value Run Value    Unit                Type
----------------------------- ----------- ----------- ------------ ------------ ------------------- ----------
 auditing                               0           0            1            1 switch               dynamic
```

3. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement for each of actions that should be audited, where *<Option>* should be substituted for the appropriate audit options (e.g. login), *<Login>* should be substituted for the appropriate login scope (e.g. all) and *<Object>* should be substituted for the appropriate object scope (e.g. all):

```
exec sp_displayaudit '<Option>', '<Login>', '<Object>'
```

4. Verify that the text returned indicates the audit setting is on.

## *4.4 Configure multiple audit tables*

Customers must configure more than one audit table and ensure that each of the tables have a threshold procedure installed. This configuration facilitates 24x7 operation. See System Admin Guide "Auditing" chapter for more information.

## *4.5 Periodically review audit settings (Level II, Not Scorable)*

**Description:**
It is recommended that the audit settings are periodically reviewed to ensure that a sufficient amount of audit events are being collected in accordance with internal and regulatory requirements and that database performance is acceptable.

**Rationale:**
Regularly reviewing audit configuration represents security best practice.

**Remediation:**
1. Carry out regular reviews of the system-wide and per server audit settings.

**Audit:**
1. Ensure that there is a record of current and previous system-wide and per server audit settings that includes change control information.

## *4.6 Review audit queue size (Level I, Scorable)*

**Description:**
Sybase ASE allows the number of audit records held in memory to be set via the `audit queue size configuration` parameter.

The default value is 100 audit records (approximately 42K of memory). If an attacker is able to trigger a crash while an audit record is stored in memory but has not been written to disk, the audit record will likely be lost (it may, however, be stored in a crash dump depending on the system configuration).

It is recommended that this setting is reviewed; the default value of 100 is likely to be sufficient for most organizations although depending on the nature of the data stored in the database, this value could be reduced.

It should be noted that decreasing this value is likely to have an effect on performance, especially on a system that is under heavy use and that generates a significant number of audit events.

**Rationale:**
If this value is set high, an attacker may be able to cover their tracks by triggering a crash.

**Remediation:**

1. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement to set the audit queue size to 100 (modify 100 as per your organization's requirements):

```
sp_configure 'audit queue size', 100
```

**Audit:**

1. Connect to the ASE server (the `sso_role` is not required) and execute the following SQL statement:

```
sp_configure 'audit queue size'
```

2. The `Config Value` and `Run Value` returned should be 100 (or your organization's requirements):

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 audit queue size                      100         104          100          100 number               dynamic
```

## *4.7 Review suspend audit configuration when device is full (Level II, Scorable)*

**Description:**
Sybase ASE is configured by default to suspend auditing when the device is full. This is controlled via the `suspend audit when device full` configuration parameter. `suspend audit when device full` is enabled by default.

If this option has been disabled (i.e. database operations continue when the audit device is full), older events will be overwritten which could allow an attacker to mask evidence of an attack.

Note that this is a potentially disruptive setting as it will suspend the audit process and all user processes that cause an auditable event when the audit device is full. To resume

normal operation, an administrator with the `sso_role` must log in and set up an empty table as the current audit table.

It is advised that this configuration is enabled for databases where maintaining an accurate audit trail is more important than the database availability. If this setting is enabled, it is recommended that audit device resources are checked regularly.

**Rationale:**
Enabling this configuration will ensure that an attacker cannot simply overwrite audit logs by submitting a large number of events.

**Remediation:**

1. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statement:

```
exec sp_configure 'suspend audit when device full', 1
```

**Audit:**

1. Connect to the ASE server (the `sso_role` is not required) and execute the following SQL statement.

```
exec sp_configure 'suspend audit when device full'
```

2. The `Config Value` and `Run Value` returned should be 1 indicating auditing will be suspended when the audit device is full:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 suspend audit when device full           1           0            1            1 switch               dynamic
```

## *4.8 Log successful and failed login attempt (Level I, Scorable)*

**Description:**
Sybase ASE can be configured to log successful and/or failed login attempts to the Windows Event Log via the `log audit logon failure` and `log audit logon success` configuration parameters.

It is recommended that both successful and failed login attempts are logged to the Windows Event Log in addition to the standard Sybase audit trail.

**Rationale:**
Logging key events such as successful and failed login attempts to multiple places (i.e. the Windows Event Log and the Sybase audit tables) means that there is less likelihood of an attacker being able to cover their tracks in the event of a compromise.

**Remediation:**

1. Connect to the ASE server as a user with the `sso_role` and execute the following SQL statements:

```
exec sp_configure 'log audit logon failure', 1
exec sp_configure 'log audit logon success', 1
```

**Audit:**

1. Connect to the ASE server (the `sso_role` is not required) and execute the following SQL statement:

```
exec sp_configure 'log audit logon failure'
```

2. The `Config Value` and `Run Value` returned should be 1:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
log audit logon failure                  0           0            1            1 switch               dynamic
```

3. Connect to the ASE server (the `sso_role` is not required) and execute the following SQL statement:

```
exec sp_configure 'log audit logon success'
```

4. The `Config Value` and `Run Value` returned should be 1:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
log audit logon success                  0           0            1            1 switch               dynamic
```

## *4.9 Monitor Usage Statistics (Level II, Scorable)*

**Description:**
Sybase ASE records statistics (CPU and I/O accounting totals) for all logins. It is recommended that these statistics are periodically reviewed. After recording and reviewing these statistics, a new accounting period should be initiated by the server (i.e. it should clear previous statistics).

**Rationale:**
The accounting totals may indicate evidence of a compromise or abuse of a user account, e.g. if an account has been compromised and is being used to exfiltrate data, this may be evident from an abnormal division of CPU workload. It is important to record previous sets

of statistics in order to be able to draw comparisons and thus determine abnormal behavior.

**Remediation:**

1. Connect to the ASE server as a user with the `sa_role` and execute the following SQL statement:

```
exec sp_reportstats
```

2. Once statistics have been recorded, a new accounting period should be initiated. Connect to the ASE server as a user with the `sa_role` and execute the following SQL statement:

```
exec sp_clearstats
```

**Audit:**

1. Ensure that there is a procedure to regularly review, record and clear server usage statistics.

# 5. Extensibility Mechanisms

This section provides guidance on securing the extensions mechanism present within Sybase that allow interaction with the operating system, network and file system resources.

## *5.1 Ensure Java is disabled (Level I, Scorable)*

**Description:**
Sybase ASE supports Java extensively, incorporating the Sun Java Virtual Machine (JVM) and offering full interoperability with Transact-SQL. Sybase implements part 1 of the SQLJ standard, and extends the standard, for instance by permitting direct references to Java methods and classes.

Java access in Sybase ASE cannot be configured on a per user basis; it is either available to all users, or to none. It is disabled by default and it is recommended that it is not enabled unless absolutely necessary. Note that only users with the `sa_role` can enable Java.

**Rationale:**
Java in ASE is a powerful target for an attacker since they can use it to interact with file system and network resources. With Java disabled, the potential for gaining a foothold on the host operating system and/or network is reduced.

**Remediation:**

1. Connect to the ASE server with a user that has the `sa_role` and execute the following SQL statement:

```
exec sp_configure 'enable java', 0
```

**Audit:**

1. Connect to the ASE server with a user that has the `sa_role` and execute the following SQL statement:

```
exec sp_configure 'enable java'
```

2. The `Config Value` and `Run Value` returned should be 0:

```
Parameter Name                Default     Memory Used Config Value Run Value    Unit                 Type
 ----------------------------- ----------- ----------- ------------ ------------ -------------------- ----------
 enable java                             0           0            0            0 switch               static
```

## *5.2 Ensure External File System Access is disabled (Level I, Scorable)*

**Description:**
Sybase ASE contains functionality for interacting with the file system through the creation of "proxy tables". This functionality is implemented by the Component Integration Service (CIS) and is accessed via standard Transact-SQL commands. It allows files and directories to be created, deleted, written to and queried.

By default only users with the `sa_role` or the `sso_role` can create proxy tables that map to files or directories. It is nonetheless recommended that external file system access is disabled.

**Rationale:**
Though an attacker would need to have compromised an account with the `sa_role` or `sso role` in order to create new proxy tables via External File System Access, this functionality, if not in use, should be disabled as a defense in depth measure. This functionality could be abused to modify operating system configuration files or create files that would allow an attacker to run code in another process.

**Remediation:**
1. Connect to the ASE server with a user that has the `sa_role` and execute the following SQL statement:

```
exec sp_configure 'enable cis', 0
```

2. If an error is returned indicating that the transaction coordinator must be disabled, execute the following SQL statement to accomplish this, restart the server and execute the above SQL statements again.

```
exec sp_configure 'enable xact coordination', 0
```

3. Restart the server.

4. Connect to the ASE server with a user that has the `sa_role` and execute the following SQL statement:

```
exec sp_configure 'enable file access', 0
```

**Audit:**
1. Connect to the ASE server (the `sa_role` is not required) and execute the following SQL statement:

```
exec sp_configure 'enable cis'
```

2. The `Config Value` and `Run Value` returned should be 0:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
 ------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 enable cis                               1           0            0            0 switch               static
```

3. Execute the following SQL statement:

```
exec sp_configure "enable file access"
```

4. The `Config Value` and `Run Value` returned should be 0:

```
Parameter Name                 Default     Memory Used Config Value Run Value    Unit                 Type
 ------------------------------ ----------- ----------- ------------ ------------ -------------------- ----------
 enable file access                       0           0            0            0 switch               dynamic
```

## 5.3 Extended Stored Procedures

Extended Stored Procedures (ESPs) are functions implemented in native code that are packaged into dynamic link libraries (DLLs). They augment the functionality provided by the database by allowing interaction with operating system and network resources such as files, the event log and email servers. ESPs have historically been a vehicle for database compromise, either by abuse of their intended purpose or via implementation flaws such as buffer overflows.

### *5.3.1 Remove operating system related ESPs (Level II, Scorable)*

**Description:**
Sybase ASE installs a number of powerful ESPs that allow interaction with the operating system. A common target for an attacker is the `xp_cmdshell` ESP, which executes a native operating system command on the host system running Sybase ASE.

The operating system user context under which the command executes is controlled by the `xp_cmdshell context` configuration parameter. Though by default, this is set to only permit execution by users with System Administration privileges at the operating system level, it should be noted that this is insufficient since an attacker who compromised an account with the `sa_role` could reconfigure the configuration parameter so that `xp_cmdshell` executes commands under the user context that the database server itself is running as.

By default, execution of the `xp_cmdshell` ESP is restricted to users with the `sa_role`. It is recommended that it is removed, along with the other operating system related ESPs; `xp_freedll`, `xp_logevent` (Windows only) and `xp_enumgroups` (Windows only). Furthermore the library that houses each of these ESPs, `sybsyesp.dll` (Windows) or `sybsyesp.so` (Unix), should be deleted from the file system to prevent them from being recreated by an attacker.

**Rationale:**
The `xp_cmdshell` ESP provides a clear path for privilege escalation from the database to the operating system. An attacker could use this functionality in conjunction with a SQL injection attack to gain a foothold on the database host using it as a launch pad to compromise other systems. If this ESP is not used, it is prudent to therefore remove it.

**Remediation:**

1. Connect to the ASE server with a user that has the `sa_role` and execute the following statements:

```
exec sp_dropextendedproc 'xp_cmdshell'
exec sp_dropextendedproc 'xp_freedll'
```

   In addition, the following statements should be executed on Windows systems:

```
exec sp_dropextendedproc 'xp_logevent'
exec sp_dropextendedproc 'xp_enumgroups'
```

   If the above statements return "Access is denied", stop the ASE server and repeat the command.

2. On Windows systems, execute the following command from a command prompt to delete `sybsyesp.dll`. It is prudent to keep a copy of the file offline in case it needs the `xp_cmdshell` functionality needs to be restored.

```
del %SYBASE%\%SYBASE_ASE%\dll\sybsyesp.dll
```

   On Unix systems, execute the following command from a command shell (assuming the SYBASE environment variables have been set):

```
rm $SYBASE\$SYBASE_ASE\lib\sybsyesp.so
```

On Unix systems it may be necessary to stop and restart the ASE server for the changes to take effect.

**Audit:**

1. Connect to the ASE server as a user with the `select` permission on `sybsystemprocs.dbo.sysobjects` (e.g. a use with the `sa_role`) and execute the following SQL statements:

```
use sybsystemprocs
go
select * from sysobjects where type='XP' and name='xp_cmdshell' or
name='xp_freedll' or name='xp_logevent' or name='xp_enumgroups'
```

2. The above SQL query should return no rows.

3. On Windows systems, execute the following command from a command prompt to verify that `sybsyesp.dll` has been deleted:

```
dir %SYBASE%\%SYBASE_ASE%\dll\sybsyesp.dll
```

   On Unix systems, execute the following command from a command shell (assuming the SYBASE environment variables have been set):

```
ls $SYBASE\$SYBASE_ASE\lib\sybsyesp.so
```

4. The above statement should return "File Not Found" (Windows) or "No such file or directory" (Unix).

### *5.3.2 Remove mail related ESPs (Level II, Scorable)*

**Description:**
On Windows systems, Sybase ASE installs a number of powerful ESPs that allow access to email via the Adaptive Server inbox. These are `xp_sendmail`, `xp_readmail`, `xp_deletemail`, `xp_findnextmsg`, `xp_startmail` and `xp_stopmail`.

By default, execution of these ESPs is restricted to users with the `sa_role`. It is recommended they are removed as a defense in depth measure if they are not in use. Furthermore the DLL that houses each of these ESPs, `sybmail.dll`, should be deleted from the file system to prevent them from being recreated by an attacker.

**Rationale:**
The email ESPs provide an attacker with suitable privileges additional means of communicating with other systems on the network and exfiltrating data. Given that ESPs have previously had a number of associated security flaws it is prudent to remove those that are not in use.

**Remediation:**

1. Connect to the ASE server with a user that has the `sa_role` and execute the following query:

```
exec sp_dropextendedproc 'xp_sendmail'
exec sp_dropextendedproc 'xp_readmail'
exec sp_dropextendedproc 'xp_deletemail'
exec sp_dropextendedproc 'xp_findnextmsg'
exec sp dropextendedproc 'xp startmail'
exec sp_dropextendedproc 'xp_stopmail'
```

2. From a command prompt execute the following command to delete `sybsyesp.dll`:

```
del %SYBASE%\%SYBASE_ASE%\dll\sybmail.dll
```

3. If the above statement returns "Access is denied", stop the ASE server and repeat the command.

**Audit:**

1. Connect to the ASE server as a user with the `select` permission on `sybsystemprocs.dbo.sysobjects` and execute the following SQL statements:

```
use sybsystemprocs
go
select * from sysobjects where type='XP' and name='xp_sendmail' or
name='xp_readmail' or name='xp_deletemail' or name='xp_findnextmsg' or
name='xp_startmail' or name='xp_stopmail'
```

2. The above SQL query should return no rows.

3. From a command prompt execute the following command to verify that `sybmail.dll` has been deleted:

```
dir %SYBASE%\%SYBASE_ASE%\dll\sybmail.dll
```

4. The above statement should return "File Not Found".

# 6. Host and Network Deployment

The section provides guidance on configuring the host, network and environment for a secure deployment of Sybase ASE.

## *6.1 Password protect database backups (Level I, Scorable)*

**Description:**
The Sybase ASE server allows passwords to be set on database backups carried out by the `dump database` command.

In addition to storing database backups in a folder with an appropriately restrictive ACL, a password should be set as part of a defense-in-depth measure.

**Rationale:**
This setting acts as a potential mitigation in the event of an attacker compromising a server containing database backups.

**Remediation:**

1. When executing the `dump database` command ensure a strong password is set via the `passwd` option (where *<Database>* should be substituted for the relevant database name, *<File>* for the full path to the database dump to be written and *<Password>* for a strong password used to protected the dump):

```
dump database <Database> to '<File>' with passwd = '<Password>'
```

**Audit:**

1. For the purposes of verifying that database dumps are password protected, the `load database` command should be used with the `headeronly` parameter. This parameter displays causes header information to be returned but does not load the database. Connect to the ASE server with a user that has the `sa_role`, the `oper_role` or is a database owner and execute the following statement for each database dump (where *<File>* should be substituted for the full path to the database dump):

```
load database from '<File>' with headeronly
```

2. Verify that the only partial header information is returned along with the message:
`Dump is password-protected, a valid password is required.`

## *6.2 Ensure the server is physically secure (Level II, Not Scorable)*

**Description:**
The Sybase ASE server should be in located in a secure environment to prevent unauthorized physical access to the machine.

**Rationale:**
It is generally accepted that physical access to a system results in its compromise even if the attacker has been granted no privileges or permissions on the target system.

**Remediation:**
1. Follow best practice recommendations for physical security and physical access control.

**Audit:**
1. Verify that the physical security and physical access control meets an acceptable standard.

## *6.3 Install on a dedicated server (Level I, Not Scorable)*

**Description:**
Sybase ASE server should be installed on a dedicated system that does not provide additional services such as web or mail and does not operate as a Domain Controller.

**Rationale:**
Vulnerabilities in other services could lead to the compromise of the Sybase ASE server.

**Remediation:**
1. Remove/disable other services.

**Audit:**
1. Verify that the only service the server provides is Sybase ASE.

## *6.4 Maintain an inventory of all ASE instances (Level I, Not Scorable)*

**Description:**
It is recommended that an inventory is kept of all versions of Sybase ASE in your organization including their corresponding patch levels.

**Rationale:**
Attacks will often target legacy servers that have not been kept up-to-date with patches.

**Remediation:**
1. Ensure that when new ASE servers are installed they are added to the inventory.

**Audit:**
1. Periodically cross-check the inventory against results from software asset management packages used in your organization.

2. Periodically perform network sweeps to identify listening Sybase ASE servers that are not present in the inventory.

## 6.5 Ensure ASE server names do not disclose sensitive information (Level I, Not Scorable)

**Description:**
When naming ASE server instances, ensure that no reference is made to version numbers or other sensitive information.

**Rationale:**

Version or other sensitive information in the server name makes it easier for an attacker to develop an attack strategy against the server.

**Remediation:**

1. When configuring ASE instances, follow a naming convention that does not include version numbers or other sensitive information.

**Audit:**

1. Ensure no ASE instances include information deemed sensitive.

## 6.6 Remove sample databases if installed (Level I, Scorable)

**Description:**
The Sybase ASE installer does not install sample databases by default. If they have been installed they should be removed.

**Rationale:**
Removal of sample databases is in accordance with the best practice principal of attack surface reduction.

**Remediation:**

1. Connect to the ASE server as a user with the `sa_role` and execute the following SQL statements to determine which sample databases are present:

```
use master
select name from sysdatabases where name = 'pubs2' or name = 'pubs3' or
name = 'images' or name = 'jpubs' or name = 'interpubs'
```

2. Execute the following SQL statement for each database name returned in the query (where *<Database>* should be substituted for the appropriate database name):

```
drop database <Database>
```

**Audit:**

1. Connect to the ASE server as a user with the `sa_role` and execute the following SQL statements to determine which sample databases are present:

```
use master
select name from sysdatabases where name = 'pubs2' or name = 'pubs3' or
name = 'images' or name = 'jpubs' or name = 'interpubs'
```

2. No records should be returned.

## *6.7 Create separate partitions for programs and data (Level I, Not Scorable)*

**Description:**
It is recommended that separate partitions are created for:

- Operating system and Sybase ASE program files
- Databases and transaction logs

**Rationale:**
Separate partitions provide greater protections via file permissions at the volume level as well as allowing greater control over data storage usage and monitoring of the database.

**Remediation:**

1. During the install process, configure the server to create system databases (e.g. master) on a separate partition to the Sybase ASE and Windows application binaries and configuration files.

2. Ensure that subsequent databases are also created on a separate partition.

**Audit:**

1. Verify that all databases are stored on a partition separate from the Sybase ASE and Windows application binaries.

## *6.8 Run a host and/or network-based packet firewall (Level II, Not Scorable)*

**Description:**
Sybase ASE can be configured to listen on a variety of network transports. By default it will listen on TCP and named pipes. Though the default TCP port is 5000, if there are multiple server instances running on a single host, there will be multiple listening ports. Dynamic listeners can also be set up via the `sp_listener` stored procedure.

It is recommended that a host and/or network-based firewall is configured to limit access to the database server port. The default Windows firewall present on Windows XP and above may be sufficient depending on your organization's requirements. Otherwise a solution with greater configurability and auditing capabilities is recommended.

**Rationale:**
It represents security best practice to segregate hosts on the network by role. Furthermore it is prudent to use firewalls, both to protect the database servers from the rest of the

network, and to protect the rest of the network from the database servers in the event of a compromise.

**Remediation:**

1. Run a host and/or network-based packet firewall to limit access to the database server port based on IP address.

**Audit:**

1. Verify that the firewall rules are suitably restrictive.

## *6.9 Harden host operating system (Level I, Scorable)*

**Description:**
The host operating system should be securely configured, disabling unnecessary services, ensuring ACLs on resources such as files, directories and network shares as restrictive as possible and ensuring it is up-to-date with relevant patches. A patching process should be in place to ensure operating system patches are applied in a timely manner.

**Rationale:**
Although the database host is likely to be located on the Intranet it may have applications connecting to it from DMZs and partner networks in addition to the threat of a malicious user that has valid albeit low privileged domain credentials. Hardening the operating system will serve the purposes of making it harder for an attacker to compromise the data within the database via an operating system attack and also harder for an attacker to fully compromise the host from the database itself.

**Remediation:**

1. Follow the guidance in the relevant CIS benchmark for the host operating system.

**Audit:**

1. Follow the guidance in the relevant CIS benchmark for the host operating system.

**Additional References:**

1. Operating system benchmarks are available at http://www.cisecurity.org/.

## *6.10 Ensure restrictive permissions on the Sybase directory (Level I, Scorable)*

**Description:**
During installation, the Sybase ASE installer will prompt to create the designated install directory if it does not exist. On Windows systems the default installation path is `c:\sybase`. The newly created folder will inherit the NTFS permissions of the parent folder.

Installing Sybase ASE to the default directory typically permits standard authenticated users to read all files in the directory, potentially exposing sensitive information if the databases themselves are located there. Furthermore, the default permissions also allow a standard authenticated user to write new files in the directory. Note that users would need sufficient privilege to logon to the system locally or remotely via Terminal Services.

The NTFS permissions on the installation directory should be reviewed and modified if necessary ensure that only Administrators and the user that the Sybase ASE server is configured to run as (typically `NT AUTHORITY\SYSTEM`) have full control of the directory and that all others users have no permissions on the folder.

Note that removing permissions for a particular user is likely to impact that user's ability to run standard database connectivity tools such as isql.exe. It is recommended that modifications to permissions are extensively tested on non-production systems first.

**Rationale:**
Weak NTFS permissions may allow a standard domain user to access sensitive data or elevate privilege.

**Remediation:**

1. Open a command prompt and execute the following command to remove standard user access to the Sybase directory. Repeat this command specifying appropriate usernames/groups until only Administrators and the user that the Sybase ASE server runs as have permissions on the directory.

```
cacls %SYBASE% /E /R "BUILTIN\Users"
```

2. The above command should return "processed dir: C:\sybase" (or an appropriate path if Sybase is installed elsewhere). If an error is returned, consult the cacls documentation.

**Audit:**

1. Open a command prompt and execute the following command to retrieve the NTFS permissions on the Sybase directory:

```
cacls %SYBASE%
```

2. Ensure that only Administrators and the user that the Sybase ASE server runs as have full control of the folder and that no other users have permissions on the folder.

## *6.11 Keep up-to-date with Sybase security patches (Level I, Scorable)*

**Description:**
Sybase ASE suffers from security vulnerabilities as any large software product invariably does. Previous publicly documented vulnerabilities have allowed for low privileged users to execute arbitrary code in the context of the operating system user that the ASE server is running under.

Updates to Sybase ASE come in the following forms:

- Emergency Bug Fixes (EBFs) are released to correct the flawed component. The accompanying documentation will typically state the severity of the issue (e.g. Sybase views this as a mandatory correction that you should implement immediately).

- Electronic Software Distribution packages (ESDs) are released periodically and typically contain multiple EBFs and other non-security bug fixes packaged as a single download, but no additional features. The most recent ESD for a given release represents the most up-to-date stable version.

- Interim Releases (IRs) are minor releases that introduce new features and enhancements, incorporating previous ESDs.

- Notification of upcoming patches and possible security threats from third party components is often announced as an Urgent Notice. Urgent Notices may be downloaded from the Sybase support site.

Occasionally details of vulnerabilities for which no patch exists may surface on security mailing lists such as Bugtraq or Full Disclosure. It is therefore recommended that these lists are regularly monitored. It may be preferable to set up keyword filters for "Sybase" since these lists carry a high volume of traffic.

EBFs, ESDs and IRs should be installed in a timely manner subject to your organization's patching policy and only after they have been fully tested on non-production servers.

**Rationale:**
It is important to keep up-to-date with patches to ensure the security and integrity of the data within the database. Privilege escalation vulnerabilities could be used directly via low privileged users or indirectly via application flaws such as SQL injection to compromise the database and gain a foothold on the host operating system.

**Remediation:**
1. Download and install the latest EBFs/ESD/IR from the Sybase download site.

**Audit:**
1. Connect to the ASE server (a privileged role is not required) and execute the following SQL query to retrieve the version number and the ID of the most recently applied EBF:

```
select @@version
```

2. Ensure the version number matches the latest IR and the most recent EBF/ESD is applied via cross-checking with the information provided at the Sybase download site.

**Additional References:**

1. An online archive of the Bugtraq mailing list is available at http://www.securityfocus.com/archive/1.
2. An online archive of the Full Disclosure mailing list is available at http://seclists.org/fulldisclosure/.
3. The Sybase support site is located at http://www.sybase.com/support/techdocs.
4. The Sybase download site is located at http://www.sybase.com/downloads.

## *6.12 Update the Java Runtime Environment (JRE) regularly if Java is in use (Level II, Not Scorable)*

**Description:**

Sybase ASE supports interaction with Java through standards such as JSQL. Sun Microsystems JRE implementation is installed by default although a user with the `sa_role` must enable Java before it can be used in the database.

Sun Microsystems regularly ship updated versions of the JRE to resolve security issues. Whilst many of these updates address technologies that have no bearing on Sybase (such as Java applets), some updates address security flaws in core JRE classes.

If Java is enabled it is recommended that the JRE is updated periodically.

**Rationale:**

Security flaws in core JRE classes may allow a low privileged user to elevate privilege.

**Remediation:**

1. Download the latest JRE from the Java download site. It is typically most convenient to download the offline, multi-language installer.

2. Open a command prompt and execute the following command to make a backup of the existing JRE installation folder:

```
C:\>xcopy /F /E /-Y /I "%SYBASE%/_jvm" "%SYBASE%\_jvm.old"
```

3. Open a command prompt and execute the following command to delete the existing JRE installation folder. Press 'Y' to confirm deletion after thoroughly checking the path has been typed as shown below:

```
C:\>rmdir /S "%SYBASE%/_jvm"
```

4. If the above command fails, stop the Sybase ASE server and execute the command again.

5. Run the downloaded JRE installer. Select the Advanced Installation Options check box and configure the following options:

    a. Set the installation path to be equivalent to `%SYBASE%\_jvm`. Note that it is not possible to supply a path in this form (i.e. using the `%SYBASE%` environment variable); the full path must be entered instead. The `%SYBASE%` environment variable corresponds to the ASE installation directory, typically `C:\Sybase` so in this case the installation path would be `C:\Sybase\_jvm`.
    b. Deselect integration with Internet Explorer.
    c. Select installation of additional language support if required.

6. Complete the JRE installation process and restart the Sybase ASE server if it was stopped in step 4.

**Audit:**

1. Open a command prompt and execute the following command to retrieve the version of the JRE current in use:

```
C:\>%SYBASE%\_jvm\bin\java -version
```

2. Ensure that this is the most up-to-date version by cross-checking it with the Java download site.

**Additional References**

1. The Java download site is located at http://java.sun.com/javase/downloads/index.jsp.

# Appendix A: References


1. Network Intelligence India Pvt. Ltd. (2003). *Guide to Sybase Security*. Available: http://www.niiconsulting.com/innovation/Sybase.pdf. Last accessed 17 July 2009.

2. Sybase, Inc. (2009). *Sybase System Administration Guide, Volumes 1 & 2*. Available: http://manuals.info.apple.com/en_US/Enterprise_Deployment_Guide.pdf. Last accessed 17 July 2009.

3. David Litchfield, Chris Anley, John Heasman, Bill Grindlay (2005). *The Database Hacker's Handbook*. USA: Wiley.

4. International Sybase User Group (2006). *ISUG Sybase ASE FAQ*. Available: http://www.isug.com/Sybase_FAQ/ASE/index.html. Last accessed 17 July 2009.

5. vulnerabilityassessment.co.uk (2007). *Sybase Vulnerability Assessment*. Available: http://www.vulnerabilityassessment.co.uk/sybase.htm. Last accessed 17 July 2009.

6. Boss Consulting (2001). *Security in Sybase*. Available: http://www.blacksheepnetworks.com/security/resources/bossconsulting/sybase_dba/sublevels/sybase.security. Last accessed 17 July 2009.

# Appendix B: Change History

| Date | Version | Changes for this version |
|---|---|---|
| 9/1/2009 | 1.0.0 | Public release |