BLU-MAN-006-R01
Allegato 04-R00
Aggiornamento del 08/09/2026

ABAP

Italgas S.p.A - Informazioni Interne

INDICE

1 INTRODUZIONE AL DOCUMENTO 3
1.1 Contesto 3
1.2 Scopo 3
1.3 Struttura del Documento 3
2 ACCESS CONTROL & PRIVILEGE MANAGEMENT 5
2.1 Autenticazione 5
2.2 Autorizzazione 5
2.3 Protezione delle Risorse 6
3 VALIDAZIONE E SANITIZZAZIONE DEGLI INPUT 7
3.1 Path Traversal 8
4 INJECTION 10
4.1 Cross-Site Scripting (XSS) 10
4.2 SQL Injection 10
4.3 Clickjacking 12
4.4 Manipolazione dei Cookie 12
5 MISCONFIGURATIONS 14
5.1 Plugin di Terze Parti 14
5.2 Virus 14
6 LOGGING & MONITORING 14
7 CRITTOGRAFIA 16
7.1 Cifratura dei Documenti 16
8 Link utili 16

1 INTRODUZIONE AL DOCUMENTO

1.1 Contesto

Questo documento è uno degli allegati al Manuale Operativo Sviluppo Sicuro e si colloca all'interno di un articolato sistema di linee guida volte a garantire la sicurezza nello sviluppo dei prodotti commissionati da Italgas. L'obiettivo è assicurare che tali prodotti siano solidi e resistenti alle minacce informatiche. È stato redatto un documento generale che affronta l'argomento in maniera trasversale senza entrare nei dettagli specifici di ciascuna tecnologia. Il presente documento, invece, approfondisce le stesse tematiche trattate nel documento generale, focalizzandosi e contestualizzandole sulla tecnologia ABAP.

1.2 Scopo

Questo documento ha lo scopo di illustrare gli aspetti di sicurezza informatica da tenere in considerazione quando si sviluppa del codice applicativo. La sicurezza del software è un requisito fondamentale per garantire la protezione delle informazioni, la privacy degli utenti e il corretto funzionamento dell'applicazione finale.
In questo documento, le tematiche affrontate si concentrano sulle possibili problematiche di sicurezza che affliggono le odierne applicazioni. Nello specifico, si fa riferimento alla tecnologia ABAP. Per ogni aspetto di sicurezza viene fornita una descrizione, degli esempi, le corrispettive contromisure e gli accorgimenti da adottare al fine di scongiurare la presenza di vulnerabilità o bug all'interno dell'applicazione finale. Per ogni tematica trattata, l'obiettivo è quello di fornire agli sviluppatori e agli architetti una lista di problematiche di sicurezza che possono emergere come conseguenza di una programmazione non orientata agli aspetti di sicurezza.

1.3 Struttura del Documento

Nel presente documento verranno trattate diverse tematiche riguardanti il linguaggio ABAP, partendo da un *livello di astrazione* più alto andando man mano scendendo verso aspetti più pratici e tecnici. Gli argomenti discussi riguardano prettamente il linguaggio e le sue peculiarità. Pertanto, si rimanda al documento di linee guida generali per lo sviluppo sicuro che sviscera i problemi in maniera più astratta senza scendere nel dettaglio del linguaggio.

Viene successivamente proposto un capitolo relativo all'**Access Control**, presentando diversi scenari che pongono l'attenzione sull'importanza di attuare adeguati meccanismi di *controllo degli accessi* e sulle problematiche di sicurezza che può comportare una mancata o scorretta implementazione di questi.

Il concetto di Access Control viene quindi esteso da quello di **Privilege Management**. La *gestione dei privilegi* è un aspetto fortemente correlato a quello del *controllo degli accessi*: implementare una solida gestione dei privilegi significa ridurre il rischio di abusi e attacchi informatici, garantendo che solo gli utenti autorizzati possano eseguire operazioni sensibili.

Ulteriori aspetti fondamentali da tenere in considerazione nello sviluppo di software sicuro sono quelli trattati nel capitolo **Validazione e Sanitizzazione degli input**. In questo capitolo viene spiegata la necessità di attuare sempre adeguate misure di *validazione* e *sanitizzazione* sugli input provenienti da utenti e sistemi esterni, i quali sono da considerarsi non fidati.

Vengono poi logicamente trattate anche le problematiche, tramite esempi e scenari, che derivano da una mancata o scorretta validazione e sanitizzazione dei dati nel capitolo **Injection**. Queste non assumono una singola forma, bensì possono avere modalità e conseguenze molto differenti tra loro in base agli asset afflitti, all'ambiente esposto e ai privilegi con cui sono eseguite.

Allo stesso modo, evitare **Configurazioni Insicure** è fondamentale per proteggere le infrastrutture e i dati da attacchi che possono compromettere la riservatezza, l'integrità e la disponibilità delle informazioni: nell'analogo capitolo vengono presentati diversi esempi di configurazioni insicure comuni nell'ambito dello sviluppo software e quali best practices seguire per poterle evitare.

Nel capitolo **Logging e Monitoraggio** viene spiegata l'importanza di attuare adeguati meccanismi di logging al fine di garantire un monitoraggio continuo che va attuato durante tutto il ciclo di vita di un software, permettendo di individuare tempestivamente problematiche di sicurezza, oltre che funzionali, e di risolverle tempestivamente.

La crittografia è alla base di molte delle operazioni che un software si trova ad eseguire durante il suo ciclo di vita, permettendo di implementare comunicazioni sicure e proteggere i dati da accessi non autorizzati.
La **Gestione della Crittografia** costituisce un aspetto fondamentale da tenere in considerazione quando si valuta il livello di sicurezza di un software: l'utilizzo di tecniche crittografiche deprecate e non più sicure o una gestione impropria delle chiavi crittografiche possono vanificare le più robuste misure di sicurezza di un sistema.

Infine, per quanto concerne l'**Utilizzo di Codice di Terze Parti,** si rimanda al documento di linee guida generali per lo sviluppo sicuro, essendo le best practice in materia valide trasversalmente a tutte le tecnologie.

Italgas S.p.A - Informazioni Interne

2 ACCESS CONTROL & PRIVILEGE MANAGEMENT

Il controllo degli accessi è un elemento fondamentale della sicurezza informatica, che determina chi può accedere a determinati dati, applicazioni e risorse e in quali circostanze. Questo processo si basa su tecniche di autenticazione e autorizzazione, oltre che la corretta configurazione dei sistemi, per verificare l'identità degli utenti e assegnare loro i permessi appropriati.

2.1 Autenticazione

L'autenticazione nel campo della sicurezza informatica è un processo fondamentale che verifica l'identità di un utente o di un dispositivo prima di concedere l'accesso a risorse, dati o servizi protetti.
Questo meccanismo assicura che solo gli individui autorizzati possano accedere alle informazioni sensibili, prevenendo così accessi non autorizzati e potenziali violazioni dei dati.

L'autenticazione può avvenire attraverso vari metodi, come la conoscenza di qualcosa di unico all'utente (come una password), il possesso di qualcosa di unico (come un token di sicurezza o un telefono cellulare), o l'essere qualcosa di unico (come le impronte digitali o il riconoscimento del volto).
Inoltre, l'autenticazione può essere rafforzata attraverso l'uso di più fattori, noto come autenticazione multi-fattore, che combina due o più metodi indipendenti per una sicurezza incrementata.

Quando si lavora con progetti ABAP, per poter accedere agli oggetti di sviluppo presenti sul backend, è necessario fornire le proprie credenziali nell'applicazione client di front-end (ABAP Development Tool).

L'utilizzo dell'autenticazione standard comporta il caricamento in memoria di user ID e password in chiaro sul client di front-end. Questo può presentare un problema potenziale di sicurezza in quanto potrebbe essere possibile per un attaccante recuperare le credenziali dalla memoria.

SAP consiglia, oltre che attivare la funzione **Secure Network Communication (SNC)**, di abilitare il meccanismo di **Single-Sign-On (SSO)** fornito da SAP NetWeaver. L'utilizzo di questo meccanismo permette di evitare di inserire le credenziali per l'autenticazione, superando il problema di sicurezza precedentemente presentato.

2.2 Autorizzazione

L'autorizzazione nel contesto della sicurezza informatica è un processo fondamentale che determina quali risorse o azioni sono accessibili a un utente, basandosi sulle sue credenziali.
Questo meccanismo assicura che solo gli utenti con i permessi adeguati possano accedere a dati sensibili o eseguire operazioni critiche all'interno di un sistema.

L'autorizzazione è spesso gestita attraverso politiche di sicurezza che definiscono chi può fare cosa, e in quali circostanze, garantendo così che ogni tentativo di accesso sia controllato e conforme alle regole stabilite. In combinazione con l'autenticazione, che verifica l'identità di un utente, l'autorizzazione è essenziale per proteggere le informazioni e le infrastrutture informatiche da accessi non autorizzati e potenziali minacce.

L'applicazione client di ABAP (ABAP Development Tool) utilizza un protocollo di comunicazione basato su chiamate HTTP con architettura REST per interagire con il backend.
Ogni oggetto di sviluppo presente sul backend è identificato da un URI. Quando tramite il client viene richiesto l'accesso ad una risorsa, la corrispondente URI viene risolta da un *router,* il quale trasferisce la richiesta del client al *controller* che detiene la risorsa richiesta.

La registrazione del controller per una risorsa avviene generalmente utilizzando una **Business Add-Ins (BAdIs)**. È possibile che venga introdotta una backdoor durante la registrazione dei controller delle risorse. Questo può succedere quando vengono registrati controller con BadI che permetto la gestione di URI randomici e pertanto anche potenziali URI malevoli.

Per mitigare il rischio associato alle backdoor, SAP NetWeaver utilizza un meccanismo di autorizzazione basato su ruoli, permettendo l'assegnazione di ruoli ad utenti del sistema da parte di un amministratore di sistema.

Tale meccanismo mette a disposizione dei ruoli predefiniti il cui utilizzo è sempre consigliato.
Ad ogni ruolo è associato uno o più oggetti di autorizzazione che determinano le capacità del ruolo.

SAP consiglia di assegnare i seguenti ruoli agli utenti utilizzati per il client (ABAP Development Tools):

- SAP_BC_DWB_ABAPDEVELOPER
- SAP_BC_DWB_DISPLAY

2.3 Protezione delle Risorse

Ogni progetto ABAP, nel client (ABAP Development Tools), fornisce una vista di tutti gli oggetti di sviluppo presenti sul backend; una copia locale di questi oggetti comprendente metadati e codice sorgente esiste anche sul front-end.

Per un attaccante potrebbe essere possibile entrare in possesso di tali metadata e codice sorgente presenti sulla macchina che ospita il client ed eventualmente manipolarli. Questi cambiamenti potrebbero essere inavvertitamente propagati sul repository ABAP, introducendo minacce di sicurezza nel sistema.

SAP raccomanda di utilizzare come workspace una cartella che non sia accessibile agli altri utenti o servizi presenti sulla stessa macchina. È fortemente raccomandato l'utilizzo del workspace di default creato in fase di installazione dell'IDE.

3 VALIDAZIONE E SANITIZZAZIONE DEGLI INPUT

La validazione e la sanitizzazione degli input sono processi fondamentali nella sicurezza informatica e nello sviluppo software. La validazione consiste nel verificare che i dati inseriti dagli utenti rispettino determinati criteri predefiniti, come formato, lunghezza e tipo di dato. Questo aiuta a prevenire errori e comportamenti imprevisti nel sistema. La sanitizzazione, invece, si occupa di pulire e modificare gli input per rimuovere o neutralizzare contenuti potenzialmente dannosi, come script malevoli o caratteri speciali, proteggendo così il sistema da attacchi come l'injection di codice. Entrambi i processi sono essenziali per garantire l'integrità e la sicurezza delle applicazioni.

SAP mette a disposizione **Web Dynpro** per lo sviluppo di interfacce grafiche (UI) in ambiente ABAP. Tale strumento permette di eseguire una validazione degli input provenienti dagli utenti.
Web Dynpro esegue in maniera automatica *controlli di esistenza* e *check sulla lunghezza degli input*; inoltre, esegue una *canonicalizzazione* degli input, trasformandoli in rappresentazioni analoghe più compatte. Infine, Web Dynpro esegue una validazione sul tipo dell'input, assicurandosi che il tipo di input ricevuto corrisponda a quello aspettato.

Se non si utilizza Web Dynpro, allora è necessario implementare manualmente anche i controlli offerti in maniera automatica da Web Dynpro. Segue un esempio di codice ABAP che implementa una funzione di validazione per un campo di tipo stringa:

```
DATA: input_string TYPE string,
      whitelist TYPE TABLE OF string WITH DEFAULT KEY.

whitelist = VALUE #( ( 'valore1' ) ( 'valore2' ) ( 'valore3' ) ).

" Controllo di esistenza
IF input_string IS NOT INITIAL.
  " Controllo sulla lunghezza
  IF strlen( input_string ) <= 255.
    " Canonicalization
    CALL FUNCTION 'HR_IT_STRING_CANONICALIZATION'
      EXPORTING
        input  = input_string
      IMPORTING
        output = input_string.

    " Validazione contro una whitelist
    IF input_string IN whitelist.
      " Input valido
      WRITE: / 'L"input è valido e accettato.'.
    ELSE.
      " Input non valido
      WRITE: / 'L"input non è nella whitelist.'.
    ENDIF.
  ELSE.
    WRITE: / 'L"input supera la lunghezza massima consentita.'.
  ENDIF.
ELSE.
  WRITE: / 'Nessun input fornito.'.
ENDIF.
```

Viene lasciato allo sviluppatore l'onere di definire funzioni di validazione su *range di valori* (ad esempio, definire il minimo ed il massimo intero possibile per un campo numerico), *whitelist* di valori ammessi (ad esempio, valori alfanumerici per un campo di tipo stringa) o *blacklist* (liste di valori non ammessi).

Nota: l'utilizzo di blacklist è sempre sconsigliato a favore di whitelist e va impiegato solo come fallback nei casi in cui l'alternativa sia non attuare nessuna validazione.

3.1 Path Traversal

Il Path Traversal, noto anche come Directory Traversal, è un tipo di attacco informatico che mira ad accedere a file e directory che si trovano al di fuori della cartella radice del web.
Un attaccante sfrutta le variabili che fanno riferimento ai file con sequenze di tipo "dot-dot-slash (../)" o utilizzando percorsi assoluti dei file, per accedere a file arbitrari memorizzati nel sistema, inclusi codici sorgente dell'applicazione, file di configurazione e file di sistema critici.
Questo attacco può essere prevenuto attraverso varie misure di sicurezza, come la validazione dell'input dell'utente, l'uso di politiche di accesso al codice e l'impiego di ambienti chrooted.

I programmi SAP che utilizzano il comando ABAP *OPEN DATASET* potrebbero essere soggetti a questa tipologia di attacco. Per poter mitigare il rischio di path traversal associato a questi programmi, è necessario che lo sviluppatore utilizzi la funzione ABAP *FILE_VALIDATE_NAME* per assicurare che il file al quale si sta facendo accesso sia in una directory valida.
È importante, inoltre, che funzionalità di accesso ai file non siano basate su input provenienti da utenti.

```
DATA: lv_filename TYPE string VALUE 'example.txt',
      lv_valid TYPE abap_bool.

CALL FUNCTION 'FILE_VALIDATE_NAME'
  EXPORTING
    filename        = lv_filename
    operation       = 'WRITE'
  IMPORTING
    valid           = lv_valid.

IF lv_valid = abap_true.
  OPEN DATASET lv_filename FOR OUTPUT IN TEXT MODE ENCODING DEFAULT.
  IF sy-subrc = 0.
    " Il file è stato aperto con successo e qui puoi scrivere i tuoi dati.
    " ...
    CLOSE DATASET lv_filename.
  ELSE.
    " Gestisci l'errore di apertura del file.
  ENDIF.
ELSE.
  " Il nome del file non è valido.
ENDIF.
```

Un'altra funzione utile messa a disposizione da ABAP è *CL_HTTP_UTILITY=>NORMALIZE_URL()*. Questa funzione esegue la normalizzazione di un path o URL, rimuovendo ../ e ./.

È possibile individuare URL o PATH non validi attraverso la funzione *CL_HTTP_UTILITY=>IS_VALID_URL()*.

Italgas S.p.A - Informazioni Interne

4 INJECTION

L'injection è una delle vulnerabilità più comuni e pericolose in ambito di sicurezza informatica. Si verifica quando un attaccante riesce ad iniettare payload maligni in un sistema attraverso input non adeguatamente filtrati, come moduli web o query di database. Questo può portare a gravi conseguenze, come l'accesso non autorizzato a dati sensibili, la compromissione del sistema o l'esecuzione di codice arbitrario. Prevenire le injection richiede una corretta validazione e sanificazione degli input, oltre all'uso di pratiche di codifica sicura.

4.1 Cross-Site Scripting (XSS)

Il Cross-Site Scripting (XSS) è una vulnerabilità di sicurezza che permette agli attaccanti di iniettare script malevoli nelle pagine web visualizzate da altri utenti. Questi script possono eseguire azioni non autorizzate nel contesto del browser della vittima, come rubare dati sensibili, dirottare sessioni o modificare il contenuto della pagina. Esistono tre principali tipi di XSS: reflected, stored e basato su DOM.

Segue un esempio di codice ABAP vulnerabile:

```
REPORT z_xss_vulnerable.

DATA: user_input TYPE string.

PARAMETERS: p_input TYPE string DEFAULT 'Hello World!'.

START-OF-SELECTION.

user_input = p_input.

WRITE: |<html><body><h1>{ user_input }</h1></body></html>|.
```

In questo esempio, il parametro *p_input* viene preso direttamente dall'input dell'utente e inserito all'interno di un tag HTML senza alcuna validazione o codifica.

ABAP, per SAB_BASIS >= 731, mette a disposizione la funzione ESCAPE, utile per sanitizzare input provenienti da utenti che vengono utilizzati in diversi contesti come HTML, XML, Javascript, URL e CSS.

Nelle versioni precedenti (SAP_BASIS 702 e <= 720), la funzione di escape è implementata nella classe CL_ABAP_DYN_PRG.

Per maggiori dettagli ed esempi di utilizzo fare riferimento alla documentazione ufficiale: SAP Encoding Functions for AS ABAP - SAP Documentation

4.2 SQL Injection

L'SQL Injection è una vulnerabilità di sicurezza delle applicazioni web che permette a un attaccante di interferire con le query che un'applicazione fa al suo database. Questo tipo di attacco consente di visualizzare dati che normalmente non sarebbero accessibili, come informazioni di altri utenti, e in alcuni casi, di modificare o eliminare questi dati. L'SQL Injection può anche essere utilizzata per compromettere il server sottostante o altre infrastrutture di back-end. Prevenire questo tipo di attacco è cruciale per la sicurezza delle applicazioni web e richiede l'implementazione di controlli rigorosi sui dati inseriti dagli utenti.

Segue un esempio di codice ABAP vulnerabile:

```
REPORT zsql_injection.

DATA: lv_tabname TYPE tabname,
      lv_query TYPE string.

PARAMETERS: p_tabname TYPE tabname.

lv_tabname = p_tabname.
CONCATENATE 'SELECT * FROM' lv_tabname 'INTO TABLE @DATA(lt_result).' INTO lv_query.

EXEC SQL.
  EXECUTE IMMEDIATE :lv_query
ENDEXEC.
```

In questo esempio, la variabile *lv_tabname* viene concatenata direttamente nella query senza alcuna validazione o sanificazione, rendendo il codice suscettibile a SQL Injection. Un attaccante potrebbe inserire un valore malevolo in *p_tabname* per manipolare la query o eseguire comandi arbitrari.

Per proteggersi da questo tipo di minaccia, è possibile utilizzare Open SQL for ABAP messo a disposizione da SAP NetWeaver Platform, il quale fa uso dei prepared statement, permettendo di mitigare il rischio associato a questa vulnerabilità:

```
REPORT zsql_injection_safe.

TYPES: BEGIN OF ty_result,
         field1 TYPE data_type1,
         field2 TYPE data_type2,
         ...
       END OF ty_result.

DATA: lt_result TYPE TABLE OF ty_result,
      lv_tabname TYPE tabname.

PARAMETERS: p_tabname TYPE tabname.

lv_tabname = p_tabname.

IF lv_tabname IS NOT INITIAL.
  SELECT * FROM (lv_tabname) INTO TABLE @lt_result.
ENDIF.
```

In questo esempio, la query Open SQL è stata scritta in modo da evitare la concatenazione diretta di stringhe, pratica comune nelle vulnerabilità di SQL Injection, utilizzando invece la sintassi di Open SQL per specificare dinamicamente il nome della tabella, racchiuso tra parentesi. Questa è una pratica sicura in ABAP.

4.3 Clickjacking

Il clickjacking è una tecnica di attacco informatico che inganna l'utente facendogli cliccare su elementi interattivi non visibili, sovrapposti a quelli legittimi di una pagina web. Questo metodo consente agli hacker di prendere il controllo delle azioni dell'utente, come clic su link o bottoni, senza che questi ne sia consapevole. Ad esempio, un utente potrebbe pensare di cliccare su un video, ma in realtà sta autorizzando un pagamento o condividendo l'accesso al proprio account social. La difesa da tali attacchi richiede una progettazione attenta della sicurezza del sito web mediante l'implementazione di politiche di sicurezza del contenuto che limitino l'uso o le capacità degli iframe.

I seguenti framework SAP offrono una protezione contro questa tipologia di vulnerabilità:

- Web Dynpro
- Business Server Pages
- NetWeaver Business Client
- Transactions running on Internet Transaction Server

Quando una applicazione ABAP non utilizza nessuno dei precedenti framework, per mitigare il rischio associato alla vulnerabilità precedentemente presentata è necessario:

- Generare i tag *<style>* utilizzando la funzione *CL_UICS_WHITELIST_CHECK->GENERATE_FP_STYLE_TAG( )*
- Generare i tag *<script>* utilizzando la funzione *CL_UICS_WHITELIST_CHECK->GENERATE_FP_SCRIPT_TAG( )*

Le whitelist utilizzate nei metodo ABAP presentati precedentemente derivano da un servizio di sistema; quando non si utilizza SAP NetWeaver come application server, è necessario implementare tale servizio in autonomia. Il servizio implementato deve essere raggiungibile attraverso un URL strutturato come uno dei due esempi seguenti:

- *http(s)://host.domain.com(:<port>)/<serviceUrl>?parentOrigin=<URL of parent>&appl=<ownLocation.pathname>*
- *http(s)://host.domain.com(:<port>)/<serviceUrl>??parm1=value1&…&parentOrigin=<URL of parent>&appl=<ownLocation.pathname>*

Per maggiori dettagli fare riferimento alla documentazione ufficiale: Clickjacking - SAP Documentation.

4.4 Manipolazione dei Cookie

La manipolazione dei cookie in ambito di sicurezza informatica rappresenta un rischio significativo, poiché i cookie sono utilizzati per memorizzare dati personali e preferenze degli utenti durante la navigazione web.
I cookie possono essere sfruttati da attaccanti per condurre attività malevole come il tracciamento non autorizzato, il furto di identità e l'attacco man-in-the-middle. In particolare, i cookie di sessione, che mantengono attiva l'autenticazione dell'utente su un sito, se intercettati, possono permettere a un malintenzionato di impersonare l'utente e accedere a informazioni sensibili.

Allo stesso modo, i cookie persistenti, che aiutano i siti web a ricordare le informazioni e le impostazioni degli utenti per visite future, possono essere manipolati per tracciare il comportamento dell'utente su diversi siti.

Questo tipo di attività può portare a violazioni della privacy e perdite di dati. Per mitigare questi rischi, è fondamentale implementare misure di sicurezza robuste, come l'utilizzo di protocolli sicuri per la trasmissione dei cookie, la limitazione della loro durata e l'adozione di politiche di sicurezza stringenti per la gestione dei cookie.

Segue un esempio di utilizzo scorretto dei cookie:

```
Cookie: IsAdmin=no; OtherCookieData=somevalue
```

Nell'esempio precedente, il cookie *IsAdmin* viene utilizzato dall'applicazione per determinare se l'utente corrente può accedere alle funzioni amministrative. Per un attaccante è sufficiente manipolare il valore da *no* a *yes* ed elevare i propri privilegi.

Quando si implementa una gestione dei cookie è importante tenere presente le seguenti considerazioni:

- I cookie devono essere sempre memorizzati server-side. Memorizzare cookie client-side permette ad un attaccante di manipolarli facilmente.
- SAP NetWeaver mette a disposizione una gestione della sessione mediante cookie sicura e consolidata. Sono da evitare implementazioni custom quando non strettamente necessarie.
- Nei cookie non vanno memorizzate informazioni sensibili.
- Implementare timeout di sessione quando l'utente è inattivo.
- Implementare meccanismi di logut manuali.

ABAP dispone della classe CL_BSP_SERVER_SIDE_COOKIE che fornisce un metodo per la creazione, eliminazione e gestione dei cookie sul server.

5 MISCONFIGURATIONS

Le misconfigurations, traducibile con 'configurazioni errate', sono problematiche di sicurezza dovute ad una erronea o deprecata interpretazione delle funzionalità di un software in fase di sviluppo del codice.

5.1 Plugin di Terze Parti

L'installazione di plugin di terze parti può comportare diversi rischi per la sicurezza dei siti web. Questi plugin possono contenere vulnerabilità non aggiornate che i cybercriminali possono sfruttare per prendere il controllo del sito, inserire keylogger, o miner di criptovalute. Inoltre, se il creatore del plugin lo abbandona o lo vende, gli utenti potrebbero non essere informati e continuare ad utilizzare moduli insicuri. È essenziale monitorare costantemente la sicurezza dei plugin installati e assicurarsi che siano regolarmente aggiornati. Alcune piattaforme bloccano il download di plugin non più supportati, ma non possono rimuoverli dai siti esistenti, il che potrebbe causare interruzioni del servizio o altri problemi. Pertanto, è fondamentale per gli sviluppatori e i proprietari di siti web essere proattivi nella gestione dei plugin di terze parti per mantenere l'integrità e la sicurezza dei loro siti.

SAP raccomanda di scegliere con cura i plugin da installare, utilizzando solo plugin derivanti da fonti fidate e che utilizzano firme digitali.

5.2 Virus

Un virus informatico è un tipo di malware progettato per diffondersi da un computer all'altro e interferire con il funzionamento del computer. Un virus può replicarsi e attaccare altri programmi o file, danneggiando il sistema operativo o cancellando dati. I virus informatici si diffondono attraverso software infetti, allegati di posta elettronica o dispositivi di memoria infetti.
Per proteggere il proprio computer, è essenziale installare un software antivirus affidabile, mantenere aggiornato il sistema operativo e fare attenzione ai file e ai programmi che si scaricano e si eseguono. La prevenzione è la migliore difesa contro i virus, che possono causare danni significativi e perdita di dati.

SAP NetWeaver metta a disposizione la **Virus Scan Interface (VSI)**, un tool di rilevamento dei virus integrabile in applicazioni SAP ABAP e Java.

Per maggiori dettagli e configurazioni fare riferimento alla documentazione ufficiale: SAP Virus Scan Interface - SAP Documentation.

6 LOGGING & MONITORING

Il logging è un aspetto fondamentale della sicurezza informatica, poiché fornisce una traccia dettagliata delle attività di sistema che possono essere analizzate in caso di incidenti di sicurezza. Attraverso i log, gli amministratori di sistema possono identificare rapidamente e con precisione eventuali comportamenti anomali o tentativi di accesso non autorizzati, facilitando così la risposta agli incidenti e la prevenzione di future minacce. Inoltre, i log possono servire come prova legale in caso di indagini su violazioni dei dati, offrendo un supporto indispensabile nel contesto della conformità normativa.

In ABAP, l'application logging è un meccanismo che permette di registrare gli eventi di un'applicazione per facilitare l'analisi e il debugging.
Il seguente esempio di codice per l'application logging inizia con la creazione di un oggetto di log tramite la transazione SLG0, poi si utilizza la funzione *BAL_LOG_CREATE* per inizializzare il log, seguita da *BAL_LOG_MSG_ADD* per aggiungere messaggi al log. Infine, si può visualizzare il log con *BAL_DSP_LOG_DISPLAY*:

Italgas S.p.A - Informazioni Interne

```
REPORT zapp_logging_example.

DATA: gv_log_handle TYPE balloghndl.

* Inizializzazione del log
CALL FUNCTION 'BAL_LOG_CREATE'
  EXPORTING
    i_s_logtype = 'ZLOGTYPE'
  IMPORTING
    e_log_handle = gv_log_handle.

* Aggiunta di un messaggio al log
CALL FUNCTION 'BAL_LOG_MSG_ADD'
  EXPORTING
    i_log_handle = gv_log_handle
    i_s_msg      = VALUE bal_s_msg( msgty = 'I' msgid = 'ZMSGCL' msgno = '000'
                                    msgv1 = 'Questo è un messaggio di esempio per il log.').

* Salvataggio del log
CALL FUNCTION 'BAL_DB_SAVE'
  EXPORTING
    i_log_handle = gv_log_handle.

* Visualizzazione del log
CALL FUNCTION 'BAL_DSP_LOG_DISPLAY'
  EXPORTING
    i_log_handle = gv_log_handle.
```

Per maggiori informazioni e configurazioni di loggin fare riferimento alla documentazione ufficiale: [Security Logging - SAP Documentation].

Italgas S.p.A - Informazioni Interne

7 CRITTOGRAFIA

Nel contesto dello sviluppo software, l'adozione di pratiche scorrette di crittografia può compromettere gravemente la sicurezza delle applicazioni.
In ABAP, ci sono diversi errori che gli sviluppatori possono commettere se non ben informati. Ad esempio, l'utilizzo di algoritmi di crittografia obsoleti o non sicuri, come DES o RC2, può rendere i dati vulnerabili ad attacchi di brute-force.

La gestione impropria delle chiavi crittografiche, come la loro memorizzazione in luoghi insicuri o la loro trasmissione in chiaro su canali non protetti, può esporre le chiavi a intercettazioni e abusi. Un altro errore comune è l'uso di valori IV (Initialization Vector) statici o prevedibili, che possono facilitare attacchi come quelli di tipo replay.

Inoltre, la mancanza di una corretta gestione degli errori durante le operazioni di crittografia può portare a divulgare informazioni sensibili attraverso messaggi di errore dettagliati.

7.1 Cifratura dei Documenti

La cifratura dei documenti è un aspetto cruciale della sicurezza informatica e della privacy dei dati. Questa tecnica consente di proteggere le informazioni sensibili da accessi non autorizzati, garantendo che solo le persone con la chiave di decifratura possano leggere il contenuto dei documenti. In un'era in cui la violazione dei dati è un rischio costante, la cifratura aiuta a preservare la riservatezza e l'integrità dei dati, sia durante la trasmissione che nell'archiviazione.
È particolarmente importante nel contesto del GDPR, che impone rigorosi standard di protezione dei dati personali, rendendo la cifratura una misura tecnica essenziale per la conformità. Inoltre, la cifratura supporta l'autenticazione e l'integrità dei dati, assicurando che le informazioni non siano state alterate durante il transito.

In ABAP, la libreria **Secure Store and Forward (SSF)** può essere utilizzata per garantire la sicurezza dei documenti attraverso la cifratura.

Utilizzando le funzioni SSF, come SSF_KRN_ENVELOPE per la cifratura e SSF_KRN_DEVELOPE per la decifratura, è possibile proteggere i dati sfruttando la tecnologia a chiave pubblica.

Questo approccio assicura che solo i destinatari previsti possano accedere al contenuto del documento.

Per maggiori dettagli fare riferimento alla documentazione ufficiale: Secure Store and Forward Mechanism (SSF) - SAP Documentation.

8 Link utili

- **Validazione e Sanitizzazione degli input**
  - Input Validation - SAP Documentation
  - Canonicalization - SAP Documentation
- **Path Traversal**
  - Directory Traversal - SAP Documentation
  - URL Input Validation - SAP Documentation

- **Autenticazione**
  - User Authentication on Front-End Client - SAP Documentation
- **Autorizzazione**
  - Role and Authorization Concept - SAP Documentation
- **Protezione delle Risorse**
  - Resource Protection on Front-End Client - SAP Documentation
- **Cross-Site Scripting (XSS)**

  - Cross-Site Scripting (XSS) - SAP Documentation
  - SAP Encoding Functions for AS ABAP - SAP Documentation
- **SQL Injection**
  - SQL Injection - SAP Documentation
- **Clickjacking**
  - Clickjacking - SAP Documentation
- **Manipolazione dei Cookie**
  - Cookie Manipulation - SAP Documentation
- **Misconfigurations**
  - A05 Security Misconfiguration - OWASP Top 10:2021
- **Plugin di Terze Parti**
  - Installing Plug-ins from Third Parties - SAP Documentation
- **Virus**
  - SAP Virus Scan Interface - SAP Documentation

- **Logging**
  - Security Logging - SAP Documentation

- **Cifratura dei Documenti**
  - Secure Store and Forward Mechanism (SSF) - SAP Documentation