7.12.3.2 Gestione dei File

Assicurarsi che gli utenti non siano autorizzati a fornire direttamente dati a tutte le funzioni dinamiche. In linguaggi come PHP, il passaggio di dati utente a funzioni incluse dinamicamente nel codice funzioni è un grave rischio di sicurezza.
Nel caso di reindirizzamenti dinamici, i dati utente non devono essere passati. Se è richiesto dall'applicazione, è necessario adottare ulteriori controlli, che includono ad esempio: l'accettazione solo dei dati correttamente convalidati e dei relativi URL. Inoltre, è importante assicurarsi che i percorsi a directory e file siano mappati in elenchi di indici di percorsi predefiniti (assicurarsi di utilizzare tali indici).
Non inviare mai il percorso assoluto del file, utilizzare sempre percorsi relativi.
Per i file e le risorse dell'applicazione, impostare autorizzazioni di sola lettura.
L'upload dei file sul server dovrebbe essere limitato ai soli utenti autenticati e solo per alcune tipologie di file accettati. Questo controllo può essere fatto usando la seguente funzione Go che rileva i tipi MIME: func DetectContentType (data[] byte) string. I file caricati dagli utenti non devono essere memorizzati nel contesto web dell'applicazione, ma in un server di contenuti o in un database. Il percorso su file systm in cui vengono memorizzati tali file non deve avere privilegi di esecuzione. Se il file server che ospita i dati caricati dall'utente è basato su *NIX, è necessario implementare meccanismi di sicurezza come l'ambiente chrooted o montare la directory del file di destinazione come un'unità logica.

7.12.3.2.1 Sorgenti dati

Ogni volta che i dati vengono trasmessi da una fonte attendibile a una fonte meno attendibile, è necessario eseguire controlli di integrità. Ciò garantisce che i dati non siano stati manomessi e che si stanno ricevendo i dati previsti. Altri controlli includono:

- Cross-system consistency checks;
- Hash totals;
- Referential integrity;
- Uniqueness check;
- Table look up check.

7.12.3.2.2 Azioni di post-validazione (azioni aggiuntive)

- informare l'utente che i dati inseriti non rispettano i requisiti richiesti e pertanto devono essere modificati per conformarli alle condizioni richieste;
- modificare i dati inviati dall'utente lato server senza notificare all'utente di tali modifiche.

7.12.3.2.3 Sanitizzazione

Dopo aver effettuato i controlli di convalida appropriati, un ulteriore passaggio che viene in genere adottato per rafforzare la sicurezza dei dati consiste nel rimuovere o modificare i caratteri ritenuti 'pericolosi'. Le azioni più comuni di sanitizzazione sono i seguenti:

- Escaping. Nel package nativo html ci sono due funzioni usate per la sanitizzazione: una per l'escape del testo HTML e un'altra per l'HTML senza escape. La funzione EscapeString(), accetta una stringa e restituisce la stessa stringa con i caratteri speciali convertiti. (es. '<' viene sostituito con '<'). Questa funzione converte solo i seguenti cinque caratteri: <,>, &, ' e ". Viceversa c'è anche la funzione UnescapeString () per convertire da entità a caratteri.
- Rimuovere i TAG. Sebbene il package html/template abbia una funzione stripTags (), questa non è esportabile. Poiché nessun altro package nativo ha una funzione capace di rimuovere tutti i tag, l'alternativa è quella di utilizzare librerie di terze parti o copiare l'intera funzione insieme alle sue classi e funzioni private. Alcuni esempi di librerie di terze parti sono:
    - https://github.com/kennygrant/sanitize
    - Il pacchetto sanitize fornisce funzioni per la sanificazione di codice HTML e dei percorsi.
    - https://github.com/maxwells/sanitize
    - Una libreria per la sanificazione di HTML che sfrutti una white list. Semplice da usare.
    - https://github.com/microcosm-cc/bluemonday