- Utilizzare GCM (Galois Counter Mode) piuttosto che CBC/ECB. GCM è una modalità di cifratura autenticata, il che significa che dopo la fase di crittografia viene aggiunto un tag di autenticazione al testo cifrato, che sarà quindi convalidato prima della decodifica dei messaggi, assicurando il messaggio da eventuali manomissioni. CBC/ECB, invece, è una crittografia a chiave pubblica o asimmetrica, che utilizza coppie di chiavi: pubbliche e private. La crittografia a chiave pubblica è meno performante della crittografia a chiave simmetrica per la maggior parte dei casi, per cui il suo uso più comune è la condivisione di una chiave simmetrica tra due parti usando la crittografia asimmetrica, in modo da poter utilizzare la chiave simmetrica per scambiare messaggi crittografati con crittografia simmetrica. A parte AES, che è una tecnologia degli anni' 90, gli autori di Go hanno iniziato ad implementare e supportare algoritmi di crittografia simmetrica più moderni che forniscono anche l'autenticazione, come "chacha20poly1305".
- Un altro package da considerare in Go, invece dell'uso diretto di AES, è "x/crypto/nacl". La "nacl/box" e "nacl/secretbox" in Go sono implementazioni delle astrazioni di NaCl per l'invio di messaggi crittografati per i due casi di utilizzo più comuni:
    - Invio di messaggi autenticati e crittografati tra due parti utilizzando la crittografia a chiave pubblica (nacl/box).
    - Invio di messaggi autenticati e crittografati tra due parti usando la crittografia simmetrica (a.k.a secret-key).
- Si deve stabilire e utilizzare una politica e un processo per la gestione delle chiavi crittografiche, in modo tale da proteggere i dati principali più sensibili dall'accesso non autorizzato. Pertanto, le chiavi crittografiche non devono essere assolutamente esplicitate, né tanto meno codificate nel sorgente (hard coded).
- Focalizzare l'attenzione sull'impiego di algoritmi crittografici più moderni come l'implementazione "https://godoc.org/golang.org/x/crypto" piuttosto che utilizzare il pacchetto "crypto/*".
- Tutti i numeri casuali, nomi di file casuali, GUID casuali e stringhe casuali generati applicativamente, devono essere creati utilizzando un generatore di numeri casuali approvato dal modulo crittografico, soprattutto quando questi valori sono potenzialmente sensibili e soggetti ad essere indovinati. Utilizzare dunque la "crypto/rand" che, anche se più lenta della "math/rand", risulta essere molto più sicura.

7.12.3.4 Gestione degli Errori e delle Eccezioni

La gestione degli errori e il logging rappresentano una parte essenziale nella protezione dell'applicazione e dell'infrastruttura. Quando si parla di gestione degli errori, ci si riferisce all'individuazione di eventuali errori nella logica dell'applicazione che potrebbero causare il blocco del sistema, a meno che non vengano gestiti correttamente.

In Go esistono funzioni per la gestione degli errori. Queste sono: il panic, recover e il defer. Quando uno stato di applicazione è *panic*, l'esecuzione normale viene interrotta, le dichiarazioni di *defer* vengono eseguite e la funzione torna al suo chiamante. *Recover* di solito è utilizzato all'interno delle dichiarazioni di *defer* e consente all'applicazione di riacquistare il controllo su una routine di panicking e di tornare alla normale esecuzione.

D'altra parte, il logging dettagliato di tutte le operazioni e delle richieste che si sono verificate nel sistema aiuta a determinare quali azioni devono essere adottate per proteggere il sistema. Poiché gli aggressori tentano di eliminare tutte le tracce delle loro azioni cancellando i log, è fondamentale che i file di log siano centralizzati e protetti da accessi non autorizzati.

Altre azioni:

- gli sviluppatori devono assicurarsi che non siano divulgate informazioni sensibili nelle risposte di errore, nonché garantire che nessun gestore di errori rilasci informazioni (ad esempio, il debug o le informazioni sulle tracce di stack).
- Il logging deve essere sempre gestito dall'applicazione e non deve basarsi sulla configurazione del server. Tutte le registrazioni devono essere implementate da una routine master su un sistema affidabile e gli sviluppatori devono inoltre assicurarsi che i dati sensibili non siano soggetti a logging