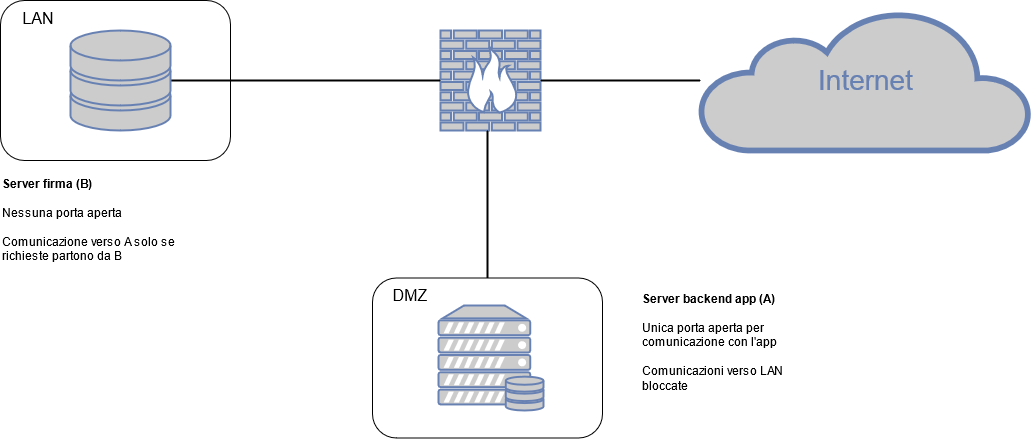
Lo storage e utilizzo delle pkey è questione complicata e controversa. Anche cercando in giro, non viene fuori nessuno schema “sicuro” e affidabile per la gestione.

Per questo motivo si cerca di avere un approccio più conservativo, applicando diversi layer di sicurezza progressivi al fine di limitare i rischi.

Nel nostro particolare caso, abbiamo bisogno di un applicativo che firmi le transazioni di mint e trasferimento dei token usando una o più pkey. Queste pkey non possono essere ovviamente conservate su una macchina esposta sulla rete, il rischio sarebbe troppo elevato.

Si può quindi utilizzare un approccio come il seguente.



Si utilizzano 2 server fisicamente separati:

1. Il server A è il server di backend che si occupa della gestione delle API per l’applicazione mobile e web.
2. Il server B è il server di firma e si occupa di firmare le transazioni e rimandarle al server A per il broadcast.

La connessione alla rete è fornita da un firewall. Questo deve essere configurato in modo da creare 2 sottoreti distinte. Una rete DMZ permette al server A di ricevere richieste dall’esterno. Ovviamente deve essere aperta verso l’esterno unicamente la porta necessaria alla comunicazione con l’app.

Una seconda rete LAN è la rete interna, posta su una classe di ip diversa e con blocco a tutte le connessioni in ingresso. Questa lan deve mettere in comunicazione il server A e il server B, ma deve permettere solo le connessioni da B verso A e non viceversa. Si può applicare un filtro sui pacchetti SYN in modo tale che vengono accettate solo le richieste provenienti da B e bloccate le altre.

1. Le richieste di trasferimento o mint, vengono generate dall’app e comunicate al server A tramite json. All’interno del JSON l’app comunica anche una versione hashata delle stesse informazioni. Per il calcolo dell’hash si utilizzano le informazioni della tx + un salt codificato come costante all’interno dell’app.
2. Il server A memorizza queste richieste in una specifica tabella del db unitamente all’hash ricevuto
3. Il server B a intervalli regolari (ex. 5sec) legge dal db su A alla ricerca di transazioni da firmare. Se la trova, recupera i dati, calcola l’hash su quei dati e lo confronta con l’hash ricevuto dall’app. In questo modo di prevengono possibili manomissioni ai dati della tx prima del mint. Se l’hash calcolata risulta diversa, la transazione viene rigettata.
4. Se l’hash è corretto, il server B firma la transazione e la scrive in un’apposita tabella del db. Questo presuppone un approccio che può essere di diversi tipi.
   1. Il server B memorizza le pkey in chiaro nel db locale.
   2. Il server B memorizza le pkey in modo cryptato nel db locale. La password per il decrypt può
      1. essere salvata localmente sul fs.
      2. Essere salvata su un archivio rimovibile
      3. Essere inserita manualmente all’avvio del daemon senza essere salvata in locale. La pw rimane in memoria fino allo spegnimento del server o l’arresto del daemon
   3. Il server non memorizza le pkey che vengono inserite a mano all’avvio del daemon. Le chiavi pertanto rimarrebbero in memoria fino allo spegnimento del server o all’arresto del daemon.
   4. Le chiavi vengono impostate come constanti in chiaro o offuscate, all’interno dello script che si occupa della firma. Lo script viene poi compilato in eseguibile in modo da dover nascondere in binario le chiavi. Questo approccio è il più promettente e sicuro, ma richiede la compilazione di un nuovo eseguibile ad ogni aggiunta di chiavi. Operazione che comunque dura pochi minuti e, entro certi limiti può essere anche automatizzata.
5. Un daemon sul server A verifica se ci sono tx firmate e se ne trova le invia alla blockchain.
6. A mining completato vengono aggiornati i dati della tx (hash, blocco ecc) e l’app riceve una notifica di avvenuto mining della richiesta.

Per aumentare l’MTFB dei daemon è possibile integrare degli alert via mail o telegram che avvisino in caso di down o errori, in modo da poter intervenire in modo rapido. È comunque sempre implementabile un servizio di auto-restart nel caso in cui i daemon non vengano ritrovati attivi dopo un certo periodo x.