algoritmi (come MD5) è matematicamente dimostrata la possibilità che la cifratura di valori testuali diversi può produrre in output lo stesso hash. Questa condizione, definita appunto collisione, può essere utilizzata da un aggressore per autenticarsi in un portale, fornendo delle credenziali di accesso differenti dalle originali.

L'attacco di brute forcing consiste nell'uso di un tool che elabora ad alta velocità combinazioni alfanumeriche col fine di intercettare chiavi crittografiche e/o password. Alcuni esempi di tool facilmente reperibili per un'oprazione di brute force attack sono: Aircrack-ng, John the Ripper, Rainbow Crack, Cain and Abel, L0phtCrack, Ophcrack, ecc.

**Contromisure**

Il brute force attack può essere contrastato bloccando l'account preso di mira, dopo un certo numero di tentativi di login falliti. Tuttavia, se l'utente malevolo ha organizzato l'attacco su un'utenza, questa potrebbe essere bloccata nuovameente, anche subito dopo lo sblocco da parte dell'help desk, determinandone la disabilitazione di fatto; se l'attacco riguarda più utenze ne può derivare un blocco del sistema (denial of service).
Bloccare l'ip dell'aggressore potrebbe portare a escludere una larga fascia di utenti leciti, in quanto l'ip potrebbe essere quello di un proxy. È preferibile bloccare un ip legandolo a un singolo device e a un singolo browser, attraverso l'uso di un device cookie.
Una misura sorprendentemente efficace è quella di utilizzare risposte imprevedibili agli attacchi brute force. Ad esempio la web application potrebbe dare codice http 200 (success) e poi reindirizzare la risposta su una pagina in cui si spiega che è in corso un brute force attack. Si può reindirizzare randomicamente l'utente su una pagina e fargli ridigitare la password.
Ogni comportamento "creativo" dell'applicazione può disinnescare gli automatismi che gli attaccanti hanno messo in opera.

### 6.3.3 Rainbow table e salt value

Una rainbow table è concettualmente una tabella in cui sono mantenuti un numero cospicuo di hash per i quali è già conosciuto il valore originario (testo in chiaro). Si possono comprare in rete svariati terabyte di tabelle rainbow, in base alla lunghezza delle stringhe trattate. Un aggressore può quindi determinare in pochi secondi l'esatta corrispondenza (clear text) semplicemente inserendo un hash nel software che gestisce le rainbow table. Questa problematica si verifica principalmente quando l'applicazione non utilizza un salt value per generare un hash. Un salt value è un fattore randomico che modifica la conformazione in output dell'hash stesso e non permette di utilizzare le classiche Rainbow Table per la relativa conversione in testo in chiaro.

Esempio: nel codice che segue, una chiave (`uncrpyptedPassword`) viene concatenata ad una stringa arbitraria (`salt`), per evitare che venga rivelata attraverso le `rainbow tables`:

```
messageDigest = MessageDigest.getInstance("SHA");
messageDigest.update((unecryptedPassword+salt).getBytes());
```

**Contromisure**

Utilizzare un valore della stringa `salt` sufficientemente lungo e complesso, in modo che le tabelle rainbow diventano completamente inutili ai fini della conversione clear text.

### 6.3.4 Archiviazione insicura

La trasmissione attraverso la rete di dati in chiaro testo o cifrati con algoritmi crittografici deboli non è l'unica pratica che può portare alla loro appropriazione indebita da parte di un aggressore. Anche archiviarli allo stesso modo nel filesystem o in un database può portare alle stesse conseguenze.