Webapp Password Storage

arnim.rupp@lhsystems.com

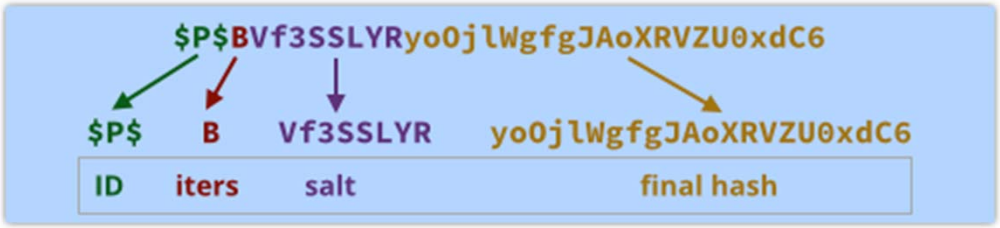
14.3.2014

Password Storage Methoden

Methode	Anwender	Runden
PHPass Portable (Salt + MD5)	Forbes	8193
PBKDF2 scrypt	OWASP Password Storage Cheat Sheet	10.000+
HMAC-SHA-256 (Salt + Key + Hashing)	OWASP Password Storage Cheat Sheet	1+
3DES (Key)	Adobe	1

Forbes: PHPass Portable

- Benutzt in Forbes Password-Dump vom Ferbuar 2014
- 6 bytes Salt
- 8192 Runden MD5



- Single Core Laptop-CPU knackt 25% der Passwörter von 500 Usern in 1 Stunde

PBKDF2 und scrypt

- Einweg-Hash mit Salt
- Konfigurierbare Rundenanzahl für die Balance, es dem Bruteforcer schwer zu machen aber trotzdem schnell Passwörter verifizieren zu können, um kein einfaches DoS-Ziel zu werden
- Nachteil:
  - Verlangsamt nur das Bruteforcen, F0rb3s66 und P@ssword1 wird weiterhin geknackt.
  - Man kommt nicht auf 0% Knackrate, die bei geschlossenen Usergruppen aber wichtig ist

Keyed functions: HMAC-SHA

- Salt + Key + Hashing
- **Vorteil:** Kein Bruteforcing möglich ohne Key
- Key außerhalb der Datenbank speichern, dadurch 100% Passwort-Sicherheit bei:
  - SQL-Injection
  - direktem DB-Zugriff (OS, Netzwerk, Backup, …)
- Angreifer bräuchte weitere Schwachstelle, um an den Key zu kommen (Local File Inclusion, Remote Code Execution, …)
- Mehrere Runden möglich um Bruteforcing bei bekanntem Key zu erschweren
- **Nachteile:**
  - Keine Änderung des Keys in einem Rutsch möglich, z.B. wenn Key abgegriffen wurde
  - Kein Wechsel das Algorithmus in einem Rutsch möglich
  - Kein Zusammenführen von mehreren User-Stämmen ohne zusätzliche Key-Zuordnung möglich

Adobes "Disaster" (Oktober 2013)

- Daten von 150.000.000 Usern geleakt, inklusive verschlüsselter Passwörter und Hints
- Gleiche Passwörter haben gleichen Ciphertext (wegen fehlendem Salt, ECB vs. CBC egal)
- Über Passwort-Hints 5-10% der Passwörter ratbar
- Geknackte Passwörter: 0
- 3DES weiter sicher
- User in Datenbank vs. /etc/shadow

Rosinenpickung

- 128 Bit Verschlüsselung = kein Bruteforcing ohne Key
- Salt = ungleiche Cipherstrings
- Mehrfaches Hashing = irreversibel + Dauer

```
[protected form] = [salt] +
        AES( key,
                    scrypt ( [salt] + [credential], 100000 )
                );
```

Vergleich

Methode	Brute Forcing nach SQLi	Reencrypt	Gleiches Password gibt gleichen Cipherstring	User zusammenlegen ohne mehrere Keys
PHPass Portable (Salt + MD5)	ja	nicht nötig	nein	ja
PBKDF2 scrypt	ja	nicht nötig	nein	ja
HMAC-SHA-256 (Salt + Key + Hashing)	nein	nein	nein	nein
3DES (Key)	nein	ja	ja	ja
AES(key, scrypt(salt, password))	nein	ja	nein	ja

Bonus

- Mit Prozedur für symmetrische Verschlüsselung kann man auch gleich die Password-Hints verschlüsseln ;)
- Key-Managment:
  - kein fester Dateiname sondern in key_a8a78d0ff555c931f045b6f448129846.php speichern => LFI können meist kein key_*.php
  - Datei Leserechte entziehen => kein Zugriff ohne Remote Code Execution (oder Race Condition)