





- Parola bilgilerini saklamaya yönelik tasarlanmış tek yönlü bir "hash" fonksiyonudur
- Diğer hash fonksiyonlarına kıyasla çok daha yavaş çalışır
- bcrypt()'in çalışması 100 ms civarı sürer
- Bu süre bir kullanıcı açısından çok uzun değildir
- Ancak bu yavaşlık topluca parolaların sınanmasını oldukça zorlaştırır



- Bu yavaş çalışmanın temelinde bir döngü içerisinde defalarca dahili olarak başka bir hash fonksiyonunu çalıştırması yatar
- Bu döngünün sayısı dışarıdan değiştirilebilir
- Bcrypt yavaş olduğu için CPU'ların hızlanması ile gündemden düşen "rainbow tabloları" yine ortaya çıkmıştır
- Bunun da önüne geçmek için bcrypt tarafından kullanıcıya özgü rastgele salt değeri kullanılmaktadır



- Böylece "rainbow tabloları"nda tutulan sık kullanılan parolaların önceden hazırlanmış hash değerleri işlevsiz hale gelmektedir
- Her kullanıcı için hash değerinin ayrı ayrı elde edilmesi gerekir
- Salt değeri anlık random bir değer olarak elde edilmektedir
- Ayrıca bu salt değeri hash değer ile birlikte tutulmaktadır



\$2a\$12\$8vxYfAWCXe0Hm4gNX8nzwuqWNuk0kcMJ1a9G2tD71ipotEZ9f80Vu





```
bcrypt(cost, salt, input)
state := EksBlowfishSetup(cost, salt, input)
ctext := "OrpheanBeholderScryDoubt"
repeat (64)
    ctext := EncryptECB(state, ctext)
return Concatenate(cost, salt, ctext)
```



İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com

