

ÖN BİLGİ

Bu belgenin resmi adresi bulunamamıştır. Alternatif adreste yedeklenmiştir.
Bu belge

- https://www.includekarabuk.com/kitaplik/indirmeDeposu/Siber_Guvenlik_Teknik_Makaleler/Teori/BaskalarinaAitMakaleler/SSL%20Say%C4%B1sal%20Sertifikalar.pdf

adresindeki makaleye çalışılarak elde edilen notlarımı kapsamaktadır. Bu çıkarılan notlar belgemde alıntılar ve/veya kişisel ilavelerim mevcuttur.

1)

Sifreleme algoritmaları

Simetrik ve asimetrik olmak üzere iki çeşittir. Simetrik anahtarlı sifreleme algoritmalarında bir veriyi sifrelemek ve sifreli veriden orjinal veriyi elde etmek için aynı anahtar kullanılır. Fakat asimetrik algoritmalarda veriyi sifrelemek ve çözmek için iki anahtar kullanılır. Veriyi sifrelemek için bir anahtar(public key), çözmek için diğer anahtar(private) kullanılır. Benim public anahtarım kullanılarak sifrelenmiş bir veri ancak benim private anahtarım kullanılarak çözülebilir. Bu yüzden asimetrik algoritmalarda açık anahtar(public key) dağıtılır, gizli anahtar(private key) saklanır. Bir de bunların haricinde mesaj özeti(message digest) olarak adlandırılan ve veri bütünlüğünü koruma amaçlı kullanılan tek yönlü çalışan hash algoritmaları vardır, md5 , sha gibi.

2)

Sayısal İmza Kavramı

Verinin içeriğinin değiştirilme durumu ve göndereninin gerçekliğini ispat söz konusu ise sayısal imza kullanılır. Sayısal imza kısaca yazılan mesajın özetinin gizli anahtar ile sifrelenmesi ve bir sıra numarası eklenmiş halidir.

Gönderici;

- i) Mesajı gönderen mesajın özetini alır
- ii) gizli anahtarını kullanarak özetini sifreler ve bir sıra numarası ekler.

Alıcı;

- i) Alıcı göndericinin açık anahtarını kullanarak sifreyi çözer
- ii) Sifre çözüldükten sonra ortaya gönderilen mesajın özeti çıkar
- iii) Aynı algoritma kullanılarak mesaj özet işlemine tabi tutulur ve doğruluğu kontrol edilir.

3)

Sertifika

Sayısal imzaları kullanarak bir verinin gerçekten beklenen kişi tarafından gönderildiği ve iletim esnasında değişimliğe uğramadığını anlayabiliriz. Peki gönderilen verinin gerçekten beklediğimiz insana ulaştığından nasıl emin olabiliriz? Yani göndereceğimiz kişinin açık anahtarını kullanarak sifrelediğimiz veriler gerçekte düşündüğümüz kişinin açık anahtarıyla mı

şifrelendi? Ve dolayısıyla gerçekte düşündüğümüz kişiye mi gidiyor? Nasıl emin olabiliriz. Burada sertifika tanımı ortaya çıkıyor. Bir sertifika basitce kişinin açık anahtarının yetkili bir sertifika otoritesi tarafından tescillenmiş halidir diyebiliriz.

Sertifika Otoritesi

Sertifika isteginde bulunan şahıs/kurumların gerçekte belirttikleri kişiler/kurumlar olduklarını doğrulayan ve onaylayan kurumdur. Verisign, Globalsign gibi.. Eger her iki taraf da ortak güvenilen bir sertifika otoritesi tarafından imzalanmış sertifika kullanıyorsa birbirlerinin public key'lerine güvenebilirler.

4)

Kendi sertifikamızı özgür bir ssl sürümü olan OpenSSL ile oluşturabiliriz. Detaylı bilgi için sayfa 3 - 13.

5)

grep huzeyfe /etc/shadow

huzeyfe:\$1\$47JagHAu\$6HvvYMo4maDHziTGdB6WT/:13465:0:99999:7::
kırmızı ve yeşilli alan huzeyfe kullanıcısının parolasının şifreli halidir. Burada kırmızı alan içerisindeki birinci ve ikinci \$ işaretleri arasındaki ifade salt olarak adlandırılır. Linux sistemlerde kullanıcı adı-parola kontrolü yaparken kullanıcının girdiği değeri alıp salt ile işleme soktukten sonra çıktığı bu dosyadaki ilgili satır ile karşılaştırıyor ve sonucu olumlu ya da olumsuz olarak dönüyor.

6)

HTTPS Nasıl Çalışır?

Testdomain.com isimli bir alan adımız olsun ve biz www.testdomain.com için yetkili bir sertifika otoritesinden(Verisign, Globalsign vs) sertifikamızı almış olalım.

İstemci ssl siteye bağlanan tarafı belirten, Sunucu ise ssl çalışan sistemi belirten taraf olsun. Bu ikisi arasındaki güvenli iletişim aşağıdaki adımlarla gerçekleşir.

İstemci sunucuya *ClientHello* olarak adlandırılan istemcinin desteklediği şifreleme algoritmaları, sıkıştırma algoritmaları gibi özellikleri belirten bir mesaj gönderir.

Sunucu bu mesajı alır ve istemciye uygun seçtiği algoritmaları vs bildiren bir mesaj gönderir. Bu mesaj *ServerHello* olarak geçer.

7)

İsim tabanlı sanal host kavramı (name-based virtual hosting) bir IP üzerinden birden fazla hostu sunma için kullanılan bilindik bir yöntem. Fakat bu yöntem eğer host edilen sunucularda SSL kullanılacaksa ise yaramıyor. Neden mi?

Bir IP üzerinden birden fazla host sunma http isteklerinde taşınan Host başlığına bağlıdır. Bir http isteği oluşturulduğunda host başlığı hedef IP'deki hangi hostu istediğini belirtir. Bizim web sunucumuza gelen istek apache (ya da başka bir web sunucu) tarafından yorumlanarak uygun site kullanıcıya gösterilir.

SSL destekli bir web sunucuda bir hosta gelen HTTPS isteginde Host headeri şifreli olarak geleceği için (OSI katmanında SSL HTTP'den öncedir) web sunucu Host alanını okuyamaz ve o alan adına özel işlemler gerçekleştiremez, dolayısı ile istenilen siteyi gösteremez. Bu yüzden virtual host ile birlikte SSL kullanılamaz. Bunun yerine Ip tabanlı sanal host kullanılmalıdır.

The image shows a Wireshark packet capture of an HTTP GET request. The packet list on the left shows a sequence of packets from 1 to 15. Packet 4 is highlighted, showing an HTTP GET request to 192.168.206.128. The 'Follow TCP Stream' window is open, showing the stream content of the selected packet. The stream content displays the HTTP request details, including the Host header: 'Host: short-home'. A red arrow points to the Host header in the stream content, with a label 'İstenilen site ismi burada belirtiliyor..'. The stream content also shows the User-Agent: 'Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)' and the Connection: 'Keep-Alive'.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.206.1	192.168.206.128	TCP	1159 > http [SYN] Seq=0 Len=0 MSS=1460 WS=0
2	0.000235	192.168.206.128	192.168.206.1	TCP	http > 1159 [SYN, ACK] Seq=0 Ack=1 Win=23360 Len=0 MSS=1460 WS=2
3	0.000253	192.168.206.1	192.168.206.128	TCP	1159 > http [ACK] Seq=1 Ack=1 Win=64240 Len=0
4	0.019798	192.168.206.1	192.168.206.128	HTTP	GET / HTTP/1.1
5	0.019828	192.168.206.128	192.168.206.1	TCP	http > 1159 [ACK] Seq=1 Ack=376 Win=6912 Len=0
6	0.019824	192.168.206.128	192.168.206.1	TCP	[TCP segment of a reassembled PDU]
7	0.019711	192.168.206.128	192.168.206.1	TCP	[TCP segment of a reassembled PDU]
8	0.019732	192.168.206.1	192.168.206.128	TCP	1159 > http [ACK] Seq=376 Ack=2921 Win=64240 Len=0
9	0.019807	192.168.206.128	192.168.206.1	TCP	[TCP segment of a reassembled PDU]
10	0.019830	192.168.206.1	192.168.206.128	TCP	1159 > http [ACK] Seq=376 Ack=4381 Win=64240 Len=0
11	0.020522	192.168.206.128	192.168.206.1	HTTP	HTTP/1.1 403 Forbidden (text/html)
12	0.020812	192.168.206.1	192.168.206.128	TCP	[TCP segment of a reassembled PDU]
13	0.020835	192.168.206.1	192.168.206.128	TCP	[TCP segment of a reassembled PDU]
14	0.023894	192.168.206.1	192.168.206.128	TCP	[TCP segment of a reassembled PDU]
15	0.024486	192.168.206.1	192.168.206.128	TCP	[TCP segment of a reassembled PDU]

İstenilen site ismi burada belirtiliyor..

Stream Content

```
GET / HTTP/1.1
Accept: image/gif, image/x-bitmap, image/jpeg, image/png, application/x-shockwave-flash,
application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, */*
Accept-Language: tr
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)
Host: short-home
Connection: Keep-Alive
```

Save As Print Entire conversation (5617 bytes) ASCII EBCDIC Hex Dump C-Arrays Raw

Close Filter Out This Stream