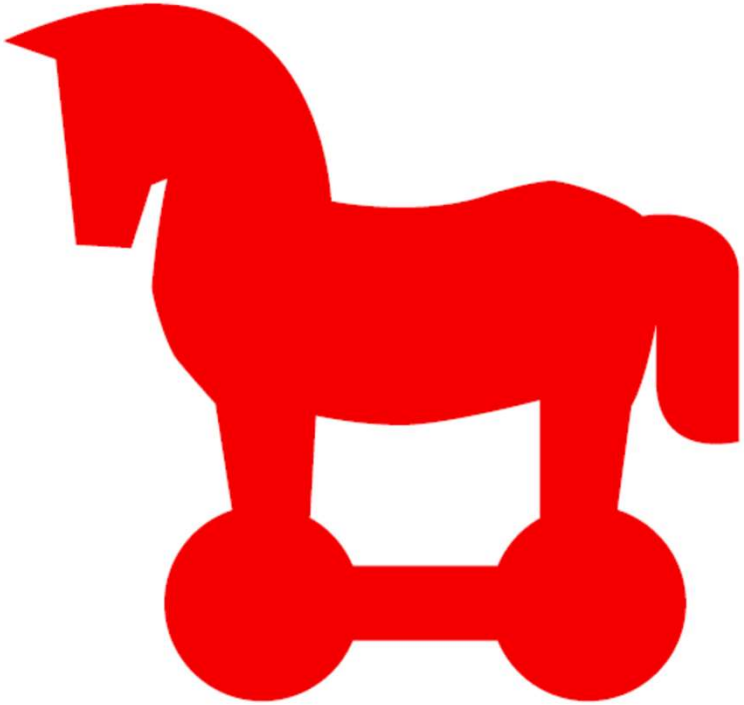
GANTEK
INTEGRATING FUTURE

GANTEK

INTEGRATING FUTURE

GANTEK
INTEGRATING FUTURE



GANTEK ACADEMY

40 YILLIK TECRÜBE İLE

Ansible ve SSH

- Ansible, ajan ve daemon kullanmadan, birden çok sistemde işleri otomatik olarak yapmayı sağlayan bir araçtır.
- Ansible, SSH ile karşı sistemde (peer) bir kabuk açar ve Python'da yazılmış modülleri yükleyerek çalıştırır.
- Sistem yöneticileri ve DevOps mühendisleri için büyük kolaylık sağlayan SSH, aslında en büyük sorun da olabilir. Sonuçta, karşı taraftaki sistemde program çalıştırılabilecek bir araç sağlıyorsunuz ve SSH iyi öğrenilip, iyi yapılandırılmadığı zaman, size en büyük kabusu yaratabilir.
- İşte bu nedenle, Ansible ve Terraform öğrenmeden önce, SSH'ın ve bu amaçla kullanılan yardımcı programların iyi düzeyde anlaşılması elzemdir.
- Aynı zamanda, şifreleme, güvenli kanal oluşturma, HTTPS, TLS/SSL, kriptografi gibi kavramların da iyi anlaşılması gerekmektedir.

GANTEK
INTEGRATING FUTURE

Şifre Güvenliği

GANTEK
INTEGRATING FUTURE

VERİ İLETİŞİMİNİN ŞİFRELENMESİ
SSH

GANTEK

Veri İletişimin Şifrelenmesi

- Ansible veya Terraform'un uzaktaki sistemlerde yapacağı işlemler SSH (Secure Shell) üzerinden gerçekleştirilmektedir.
- SSH, uzaktaki sistemde komut verebilmek için bir kabuk programı çalıştırmak amacıyla kullanılan bir yazılımdır.
- SSH, uzaktaki makinede bir kabuk açar ve uzaktan buraya komutlar verilmesini sağlar.
- İnternet veya korumalı olmayan bir ağ üzerinden gerçekleştirilen aradaki iletişim şifrelerle güvenli hale getirilir.

Simetrik şifreleme

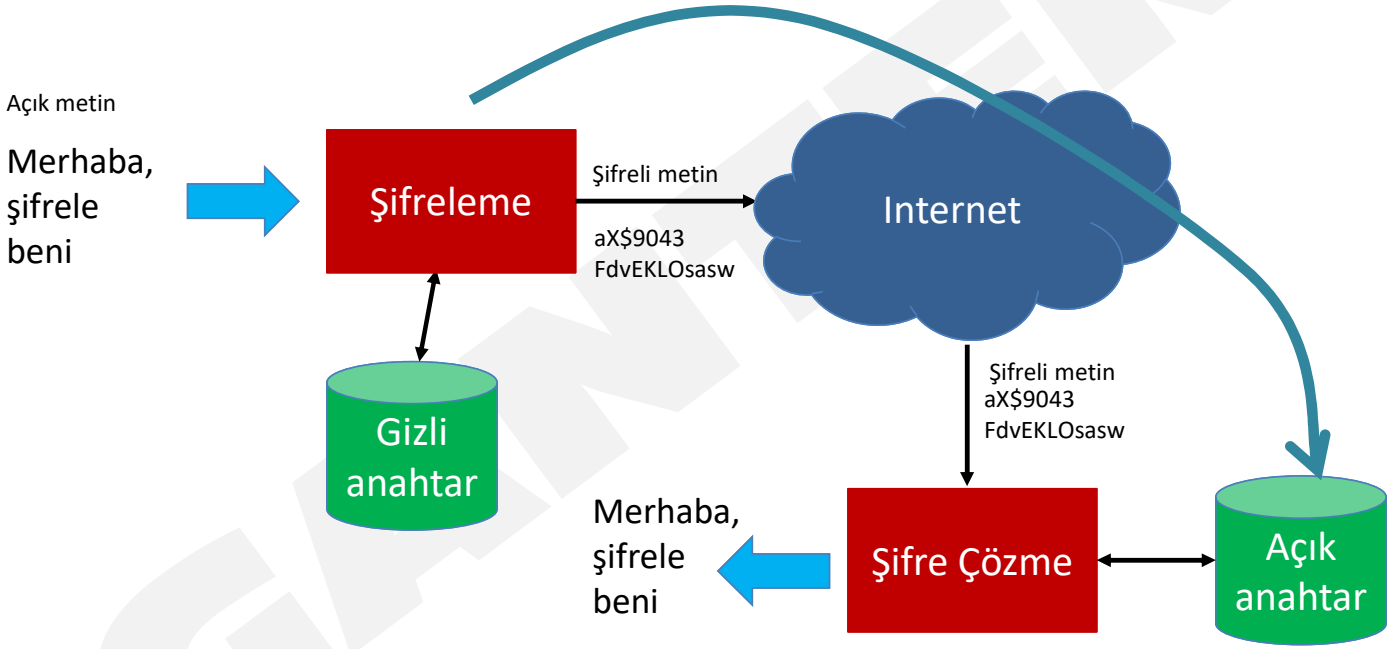
- Simetrik şifrelemede, mesajı gönderen de alan da aynı şifreyi kullanır. Her iki tarafın da şifreyi koruması gereklidir.
- Kullanılan algoritmalar şunlardır:
  - Twofish
  - Serpent
  - AES (Rijndael)
  - Camellia
  - Salsa20
  - ChaCha20
  - Blowfish
  - CAST5
  - Kuznyechik
  - RC4 (güvensiz)
  - DES
  - 3DES
  - Skipjack
  - Safer
  - IDEA

GANTEK

Asimetrik Şifreleme

- Asimetrik şifreleme, iki farklı şifrenin kullanılmasına dayan bir sistemdir ve «**public-key encryption**» olarak bilinir.
- Şifrelerden biriyle şifrelenen bir metin, diğer şifreyle açılabilir.
- Genel uygulamada, şifrelerden biri açık olarak dağıtılırken, diğeri gizlidir ve anahtarın sahibi tarafından korunur.
- Algoritmik olarak, karmaşık bir yöntemle şifreler oluşturulur ve genellikle şifre boyutu 3072-4096 bit civarındadır.
- Bu nedenle şifreleme ve şifrenin geri alınması süreçleri yavaştır.
- Güvenlik olarak diğer yöntemlere göre daha ileri seviyeli koruma sağladığı düşünülür.
- Asimetrik şifrenin kullanıldığı alan, küçük parçalar halindeki verilerin şifrelenmesi işleridir. Büyük çapta verilerin şifrelenmesi için diğer yöntemlere göre daha fazla kaynak gerekir.

Asimetrik Şifreleme



Açık şifre, iki sistem TCP/IP üzerinden bağlandıktan sonra, diğer tarafa aktarılır.

Asimetrik Şifreleme Algoritmaları

- Asimetrik şifrelerin oluşturulması ve kullanılması için farklı algoritmalar mevcuttur.
- Geliştirilen bazı algoritmalar, tahmin edilebilir şifreler ürettikleri için kullanılmamaya başlamıştır.
  - Diffie-Hellman: Asal sayılara bağlı algoritma 1977'de Diffie ve Hellman tarafından patentlenmiştir. Her iki taraf arasında bir gizli anahtarın paylaşılması esasına dayanır. «Ortadaki adam» (man in the middle) saldırılarına açıktır.
  - Rivest Shamir Adleman (RSA): 1983'te patentlenen bir algoritmadır. Asal sayıların çarpımı yöntemini kullanır. Bağlantı kurulduktan sonra ek şifrelemeyle, ortadaki adam saldırılarına karşı tedbir alınır. 4096 bitlik şifrelerle yaygın olarak kullanılan bir algoritmadır.
  - Digital Signature Algorithm (DSA): 1991'de, RSA algoritmasına alternatif olması için NSA tarafından geliştirilmiştir. RSA ve DSA Apache gibi web sunucularla yanyana kullanılabilir. DSA'in farkı, sertifikaların üçüncü bir otorite tarafından onaylanabilmesidir. RSA'ye göre daha az sayıda bitlik şifreler kullanır.
  - Elliptic Curve Cryptography (ECC), Elliptic Curve Digital Signature Algorithm (ECDSA) : Bu yöntemlere esin kaynağı olan matematiksel yöntemler 150 seneden beri üzerinde çalışılan tekniklerdir. Neal Koblit ve Victor S. Miller tarafından 1985'te önerilmiştir. 2005'te NSA tarafından yayınlanmıştır. ECC'de RSA'de olduğu gibi iki asal sayının çarpımını kullanmaz ve eliptik eğrilerde yapılan işlemlere dayanır. ECDSA = ECC + DSA (ECC'nin orijinal sürümü ve DSA'ın bir türü)

Hangisi iyi?

- Simetrik şifrelerde daha az bitle daha üst seviye güvenlik sağlanabiliyor. ECC ise RSA'ye göre daha az bitle güvenliği sağlayabiliyor.

Key Size Comparison:

Symmetric Key Size (bits)	RSA Size (bits)	Elliptic Curve Key Size (bits)
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

Recommended Key Sizes According to NIST

https://sectigo.com/resource-library/rsa-vs-dsa-vs-ecc-encryption
https://blog.cloudflare.com/why-are-some-keys-small/

RSA Algoritması

- RSA algoritması, 1977'de Ron Rivest, Adi Shamir ve Leonard Adleman tarafından geliştirilmiştir.
- RSA algoritması, iki asal sayının çarpılmasının, çarpanlarının bulunmasından daha kolay yapılabilmesi ilkesine dayanmaktadır.
- Açık şifrenin oluşturulması algoritması
  - İki asal sayı bulunur. Örneğin P=53 ve Q=59.
  - n = P × Q değeri bulunur. Bu açık şifrenin ilk parçasıdır.
  - Φ(n)=(P-1)×(Q-1)
  - Bir tamsayı e değeri seçilir. Bu değer, 1'den büyük ve Φ(n)'den küçük olmalıdır. Aynı zamanda Φ(n)'nin çarpanlarından biri olmayacaktır.
  - Açık anahtarımız n ve e'den oluşturulur.
- Gizli anahtar Φ(n), e ve herhangi bir tamsayı k değerinden oluşturulur.
  - d = (k × Φ(n) + 1)

https://www.nordugrid.org/documents/rsalabs_faq41.pdf
https://www.geeksforgeeks.org/rsa-algorithm-cryptography/

RSA Algoritması

- Örnek (Açık ve Gizli anahtarı oluşturalım):
  - P ve Q değerlerini seçelim: P = 53, Q = 59
  - n = P × Q = 3127
  - Φ(n)=(P-1)×(Q-1)=3016
  - e'yi 3 olarak seçelim.
  - Gizli anahtarı k=2 için hesaplayalım:
    - d = (k × Φ(n) + 1) = 2011
- Metindeki değerleri nümerik değerlere çevirelim. «HI» kelimesinden, «8» ve «9»dan oluşan 89 değerini oluşturalım.
  - Şifrelenmiş değer: $89^{e} \bmod n$ = 1394
  - 1394'ü de açmak için $1394^{d} \bmod n$ = 89

https://www.geeksforgeeks.org/rsa-algorithm-cryptography/
https://www.abc.net.au/news/science/2018-01-20/how-prime-numbers-rsa-encryption-works/9338876

Public-Key Infrastructure (PKI)

- PKI, «**`public-key encryption`**» yönteminde kullanılan dijital sertifikaların oluşturulması, yönetilmesi, dağıtılması ve geçersiz hale getirilmesi için gerekli roller, donanımı, yazılımı ve yordamları içeren bir yapıdır.
- RSA algoritması, açık ve gizli şifrelerle, mesajların şifrelenmesini sağlamaktadır.
- Bilindiği gibi, açık şifre özgürce dağıtılmaktadır.
- **SORU**: açık şifre alındığında, bu şifrenin hangi bilgisayara ait olduğunu nasıl bilebiliriz?
- **SORU**: açık şifreyi bir İnternet sitesine bağlayabilir miyiz?
- **SORU**: açık şifreyi gerçek bir kişiye bağlayabilir miyiz?
- **CEVAP**: Bütün bunlar PKI'ın kurulma sebepleri arasında yer alır.

PKI

- PKI sertifikaları, açık anahtar ve bu anahtara sahip bilgisayar/site ile ilgili bilgileri içeren, bir çeşit dijital pasaporttur.
- Sertifika içinde, CA (Certification Authority) adı verilen bir sistemden alınan onaylar da bulunur.

https://www.csoonline.com/article/3400836/what-is-pki-and-how-it-secures-just-about-everything-online.html

Şifre uzunluğu

- Şifre uzunluğu, şifrelemenin ne kadar iyi olduğunu belirler.
- Günümüzde, «brute force» saldırılarına önlem olarak en az 3072 bit'lik şifrelerin kullanılması gerekmektedir.
- Ancak, tedbir amaçlı 4096 bitlik şifrelerin kullanılması önerilmektedir.
- Bilgisayar hızlarının düşük olduğu yıllarda, bu kadar büyük şifrelerin kullanımı, iletişimi yavaşlattığından tercih edilmiyordu. Günümüzde büyük şifrelerin iletişim hızına etkisi azdır.

SSH Kullanımı

- SSH'ın ilk versiyonu, *Tatu Ylönen* isimli bir Finlandiyalı tarafından geliştirildi ve ücretsiz bir yazılım olarak sunuldu. Zaman içinde gelişen SSH çok sayıda kullanıcı tarafından benimsendi ve halen de kullanılmaktadır.
- Ancak, *Ylönen*, yaygınlaşan bu yazılımını ticari bir ürün hale getirdi.
- SSH programının ticari hale gelmesi, bu konudaki çalışmaları durdurmadı ve SSH bir protokol olarak IETF tarafından RFC4253 koduyla standartlaştırıldı.
- 1999'da, SSH'ın açık kaynak kodlu olduğu dönemdeki eski kaynak kodlarına dayanan, açık kaynak kodlu OpenSSH projesi başlatıldı.
- SSH'ın korsanlar tarafından kırıldığına dair bir çok söylenti (örneğin Snowden vakası) bulunmaktadır. Ancak bu konuyla ilgili ispatlanmış bir durum yoktur.

Veri iletişimini şifreleyin

GANTEK
INTEGRATING FUTURE

- Sisteme gelen veya sistemden dışarı çıkan veri iletişimi şifrelenmelidir.
- Şifrelenmeden yapılan haberleşme geçen trafiğin üçüncü kişiler tarafından dinlenmesi anlamına gelen «**sniffing**» denilen saldırılara maruz kalabilir.
  - **scp**, **ssh**, **rsync** veya **sftp** kullanın.
  - **sshfs** ve **fuse** araçlarını kullanarak ev dizinine erişin
  - **GnuPG** ve ve iletişiminizi şifrelemek ve imzalamak için kullanılan bir mekanizmadır.
  - **fugu**, güvenli dosya transferi için kullanılan komut satırından kullanılan **sftp** programının grafik arayüzüdür.
  - **Filezilla**, ftp, SSL/TLS üzerinden FTP ve SFTP protokollerini destekleyen diğer bir programdır.
  - **OpenVPN**, SSL VPN sağlayan bir yazılımdır.
  - **Lighttpd** SSL
  - **Apache** SSL
  - **FTP**, **Telnet** ve **Rlogin/Rsh** hizmetlerini kullanmaktan kaçının.

https://www.cyberciti.biz/tips/linux-security.html

«**sniffing**» nedir?

- İki veya daha fazla taraf arasında gerçekleşen şifresiz iletişimi, ağ üzerinden geçen paketlere bakarak kaydeden yazılıma «**sniffer**» denir.
- Güvenlik ön planda tutulan ağ sistemlerinde, ilgisiz kullanıcıların paketleri görmemesi için çeşitli tedbirler alırlar. Örneğin VLAN'lar, kullanıcıları böler ve ilgisiz olanların paketleri görmesi engellenir.
- Ancak, son noktada kullanılan L2 anahtarlar veya ağ paketlerinin geçtiği bir sistemi ele geçiren art niyetli kişiler, geçen paketleri kaydedebilir ve işleyerek farklı amaçlarla kullanabilir.
- Ağ üzerinde akan trafiğin SSH, SSL vb yöntemlerle şifrelenmesi, «**sniffing**» durumunu engellemektedir.

SSL ve TLS nedir?

- Secure Socket Layers (SSL) ve güncellenmiş hali TLS (Transport Layer Security), ağ üzerindeki iki bilgisayarın birbirleriyle haberleşmesini yetkilendirilmiş ve şifrelenmiş bağlarla yapmasını sağlayan protokollerdir.
- TLS, SSL'in devamı niteliğinde ve daha güncel bir protokol olmasına rağmen hala SSL olarak bilinmekte ve doğru şekilde SSL/TLS olarak ifade edilmektedir.
- Haberleşme, SSL sertifikası adı verilen ve içinde açık (public) ve gizli (private) anahtarlar (dijital şifre) içeren ve web haberleşmesi için gerekli dijital bir dokümandır.
- Şifreleme asimetriktir. Açık anahtarla şifrelenen veri, gizli anahtarla açılabilir. Tam tersi de geçerlidir. Gizli anahtarla şifrelenen veri, açık anahtarla açılabilir.
- Gizli anahtar, SSL sertifikasının sahibi tarafından korunurken, açık anahtar karşı sistemle paylaşılabilir.

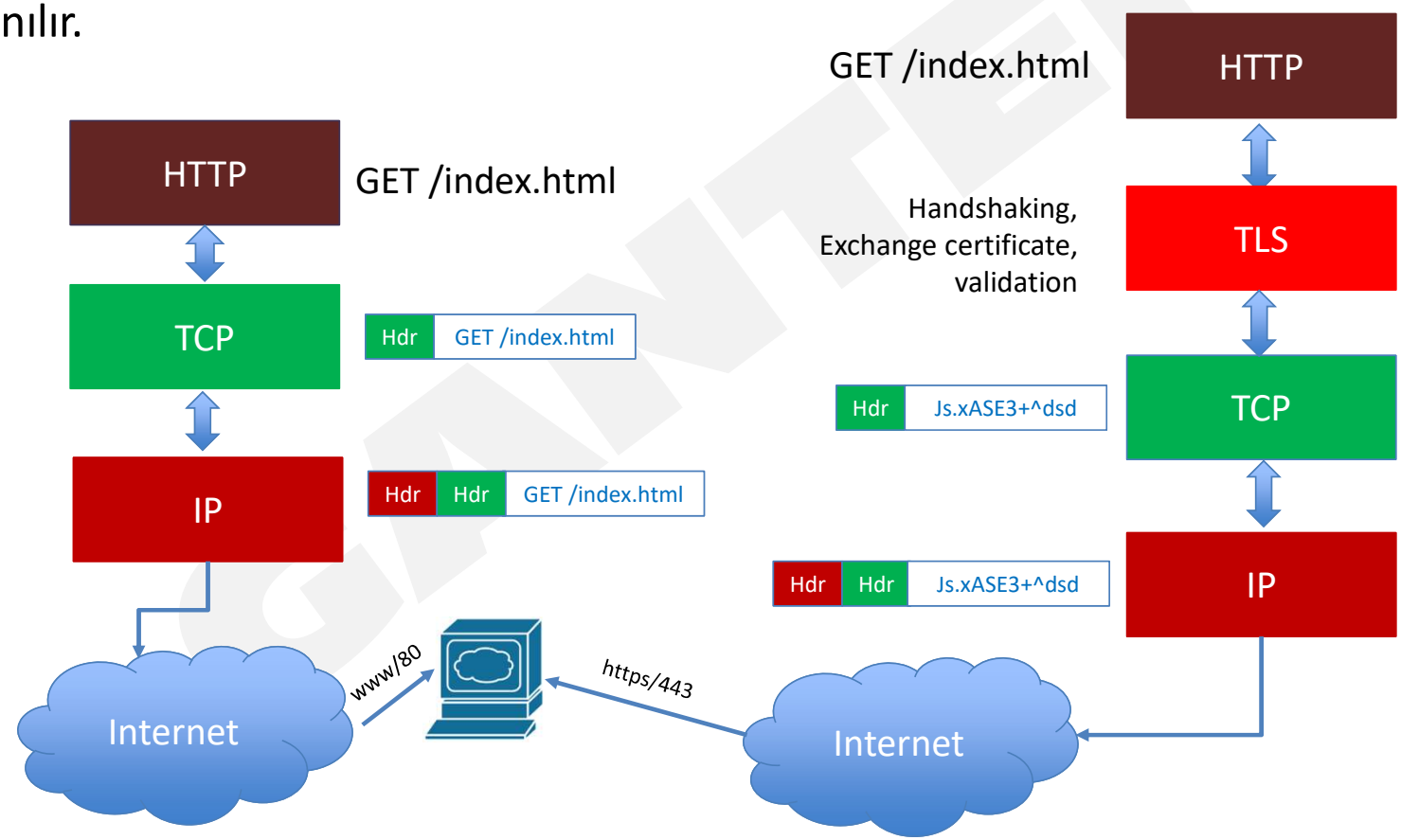
https://www.ssl.com/faqs/faq-what-is-ssl/

TLS/SSL Tarihçesi

- SSL 1.0 ilk web izleyicisi olan Netscape'in 1994'teki tasarımında yer almıştır. Ancak ayrıntılar kaybolmuştur.
- SSL 2.0, Netscape tarafından Kasım 1994'te yayınlanmış ve çeşitli açıkları bulunan bir standarttır. Zayıf şifre kullanımıyla birlikte aktarılan mesajlarda bütünlük sorunları yaratan zafiyetleri bulunmaktadır.
- Taher Elgamal, Netscape'te çalıştığı sırada SSL üzerinde çalıştığı için SSL'in babası olarak bilinir.
- SSL 3.0, Netscape ve Paul Kocher tarafından tasarlanmış ve Kasım 1996'da yayınlanmıştır.
- TLS 1.0, SSL 3.0'daki fikirleri kullanan bir protokol olarak Ocak 1999'da yayınlanmıştır. Ancak SSL 3.0 ile uyumlu değildir.
  - Protokol ilk olarak RFC 2246 adıyla sunulmuştur.
  - SSL v2.0'in RFC dokümanı 30 sayfa iken TLS v1.0'inki 80 sayfaydı.
- TLS 1.1, 2006'de çıkmıştır ve Aralık 2021'de kullanımı bırakılmıştır.
- TLS 1.2, 2008'de yayınlanmıştır (RFC 5246).
- TLS 1.3, 2018'de yayınlanmış en son sürümdür.
- Şu anda TLS 1.2 ve TLS 1.3 beraber kullanılmaktadır.
- TLS/SSL HTTP, FTP ve Telnet üzerinde çalışabilir. Ancak HTTP protokolü için optimize edilmiştir.

HTTP/HTTPS Protokolü Nasıl Çalışır?

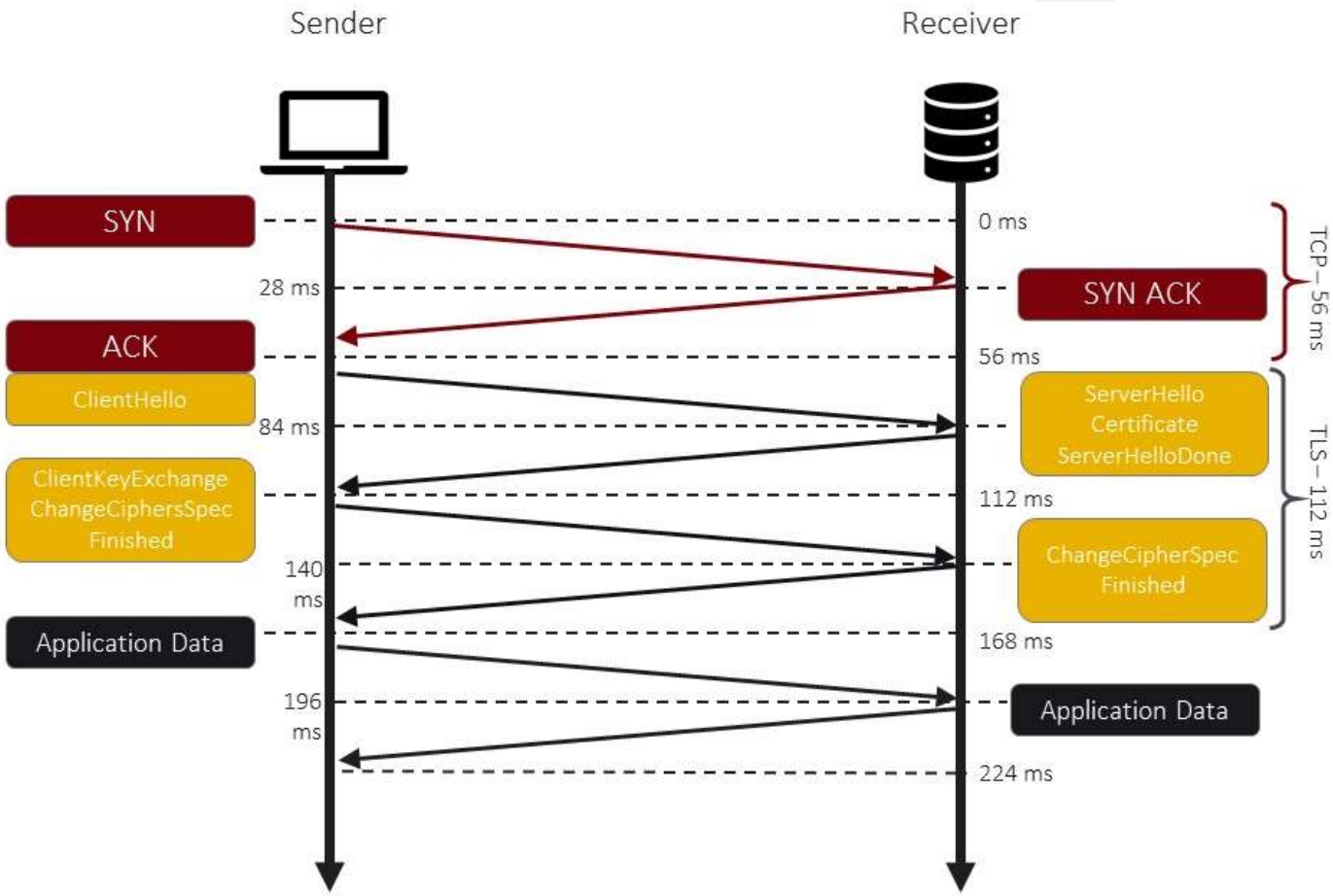
- HTTP, alttaki TCP ve IP yığıtlarını kullanarak, web sunucusuna komut gönderilmesini ve sunucudaki dosyaların bu komutlar sayesinde web izleyicisi (Chrome, Firefox, vB) tarafından alınmasını sağlayan bir protokoldür.
- HTTPS ise, transparan şekilde TLS ile TCP/IP yığıtlarına şifreli mesajları göndermek için kullanılır.



TLS temelleri

- TLS'in içinde haberleşmenin farklı aşamalarında kullanılan iki ayrı protokol bulunur:
  - El sıkışma protokolü: İstemci ve sunucu arasında açık-anahtar temelli bir kriptografi yöntemiyle, ortak gizli bir anahtar oluşturulması
  - Mesaj şifreleme protokolü: El sıkışma protokolünde oluşturulan gizli anahtar, istemci ve sunucu arasındaki iletişimi şifrelemek için kullanılır.

HTTPS



https://love2dev.com/blog/how-https-works/

HTTPS’de SSL Sertifikası

- SSL sertifikasının içinde, açık ve gizli olmak üzere iki şifre bulunur.
- Gizli şifre web sunucusunda tutulur ve açık şifre web sunucusuna bağlanan sistemlerle paylaşılır.
- Sertifikanın içinde sadece şifreler bulunmaz ayrıca web sitesinin alan adı ve sitenin sahibi hakkında da bilgi içerir.
- SSL sertifikasını kendiniz oluşturabileceğiniz gibi, alan adınız için CA (Certification Authority) denilen güvenilir şirketlerden ücreti karşılığı temin edebilirsiniz.
- CA’lerden elde edilen bir sertifika üzerinden yapılan web sitesi haberleşmesinin güvenilir olduğu kanıtlanır. Bunu işletim sistemleri ve web izleyicileri onaylar.

https://www.forbes.com/advisor/business/software/best-ssl-certificate-services/

El Sıkışma TLS v1.2

- Sunucu 443 portunu dinler.
- İstemci 443 portuna bağlanır ve **`ClientHello`** mesajını sunucuya iletir. Bu mesaj istemcinin desteklediği TLS sürümlerini, istemcinin desteklediği şifreleme yöntemlerini ve 32 baytlık rasgele bir sayıyı (**`Client Random`**) içerir.
- Sunucu «**`hello`**» mesajını, ServerHello mesajıyla birlikte gönderir. Sunucu, **`ServerHello`** mesajında, haberleşmede kullanacağı şifreleme yöntemini, TLS sürümünü, 32 baytlık rasgele sayıyı (**`Server Random`**) ve sunucunun SSL sertifikasını gönderir.
- İstemci, sunucunun kimliğini SSL sertifikası ile onaylar. SSL sertifikası aslında o sunucuya ait açık anahtarı (**`public key`**) içerir.

El Sıkışma TLS v1.2

- İstemci, sunucuya, kendi oluşturduğu 48 baytlık rasgele bir sayının (pre-master key) sunucunun SSL sertifikasındaki açık anahtarla şifrelenmiş halini gönderir. Eğer sunucu, SSL sertifikasındaki sunucuysa, bu şifreyi çözüp istemciye geri iletecektir.
- Sunucu 48 baytlık sayıyı kendi gizli anahtarı ile açar. Sunucu ve istemcinin her ikisinin elinde pre-master key, Server Random ve ClientServer bulunmaktadır. Sunucu ve istemci, ayrı ayrı, bu üç değeri kullanarak aynı simetrik bir şifreyi oluştururlar (daha önce karar verdikleri şifre yöntemini kullanarak).
  - Her iki taraf aynı simetrik şifreyi oluşturup oluşturmadığını kontrol etmelidir.
- İstemci, ChangeCipherSpec mesajını sunucuya iletir. Bu mesaj, sunucuya, oturum anahtarını oluşturacağını belirtmektedir. Bu anahtar, oturumda karşı tarafa aktarılacak mesajların şifrelenmesinde kullanılacaktır.
- İstemci, Sunucuya doğru anahtara sahip olduğunu ispatlar. İstemci, daha önceki el sıkışma mesajlarını hash'leyerek sunucuya iletir. Sunucu aynı şifreye sahipse, aynı hash mesajlarını elde edecektir.
- Sunucu istemciye aynı mesajları hash'leyerek gönderir ve istemciye aynı şifreye sahip olduğunu ispatlar.
- Bu şekilde, artık her iki tarafça kabul edilen gizli bir bağlantı kurulmuş olur.

TLS v1.3

- TLS v1.2'de karmaşık bir el sıkışma protokolü bulunmaktadır. TLS v1.3'de bu protokol kısaltılmıştır.
- El sıkışma sırasında, istemci ve sunucu destekledikleri algoritmaları birbirlerine iletmektedir. Bu şekilde her iki taraf hangi algoritmayla şifrelemeyi yapacaklarına karar vermektedir.
- Ancak geriye yönelik uyumluluk, her zaman için sıkıntılı bir özelliktir. Eski algoritmaların sorunları da yeni sürümlere yansır.
- TLS v1.3'de artık sadece güçlü şifreleme yapılan algoritmalar (örneğin ephemeral Diffie-Hellman) desteklenmektedir. 37 algoritmanın 32'si olası güvenlik sorunları nedeniyle devreden çıkarılmıştır ve iletişim için sadece 5 algoritmanın kullanılması istenmektedir. Bu şekilde, hem istemcilerin sorunlu algoritmaları seçmeleri engellenmekte hem de el sıkışma süresi azalmaktadır.

https://auth0.com/blog/the-tls-handshake-explained/

TLS v1.3

GANTEK
INTEGRATING FUTURE

- Sunucu 443 numaralı TCP portunu dinler.
- İstemci, sunucudaki 443 numaralı porta bağlanır ve 5 tane şifreleme algoritmanın her biriyle oluşturduğu şifreleri ve algoritma sürümlerini sunucuya iletir.
- Sunucu, istemciye ServerHello mesajı gönderir. Sunucu, kendi oluşturduğu anahtarları istemciye iletir. Aynı zamanda, bu anahtarla şifrelenmiş SSL anahtarını da istemciye gönderir. İstemci, bunları kullanarak kendisi de oturum anahtarını oluşturacaktır.
- İstemci, kendisi de şifreyi oluşturur ve şifrelenmiş SSL sertifikasının şifresini açar ve sorun yoksa el sıkışma tamamlanmış olur.
- TLS v1.3'de iki çift açık/gizli anahtar bulunur. Birincisi, her oturum için oluşturulan geçici (ephemeral) anahtar çifti ve diğeri de sunucunun kimliğinin onaylanması için kullanılan SSL açık/gizli anahtar çiftidir.

https://auth0.com/blog/the-tls-handshake-explained/

SSH: Secure Shell

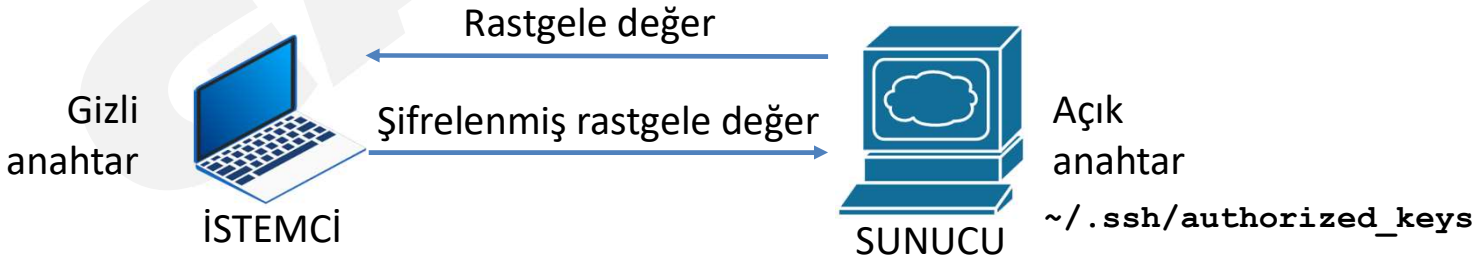
- SSH, güvenilir olmayan (örneğin İnternet) üzerinden, hedef sistem üzerindeki ağ hizmetlerine , kriptografik yöntemlerle (açık ve gizli kriptografik anahtar çiftini kullanarak) her iki uç arasında güvenli bir şekilde bağlantı kurulmasını sağlayan protokoldür.
- Açık ve gizli kriptografik anahtar çifti, CA (Certification Authority) olmadan sistemde kullanıcı tarafından oluşturulabilir.
- Açık (public) anahtar, bağlanacak sistemlere dağıtılır. Gizli anahtar her zaman için gizli tutulmalı ve sistemde saklanmalıdır.
- SSH genellikle uzaktaki makineye güvenli şekilde login yapmak için kullanılır. Ayrıca sisteme girişte login şifresi de istenebilir.
- Ancak, farklı özellikler de (tünelleme, TCP portunun **`forward`** edilmesi, uzaktaki sistemde komut çalıştırılması, sisteme şifre girmeden güvenli girişi, vb) sunmaktadır. Ayrıca, SFTP (SSH file transfer protocol) ile güvenli dosya transferi, SCP (**S**ecure **C**o**P**y) protokolleri de desteklemektedir.
- Bağlanılacak sistemde yer alan SSH sunucusuna bağlanmak için SSH istemcisi kullanılmalıdır.

SSH ve Bulut Sistemleri

- Bulut sistemlerinde yer alan sunuculara bağlantı kurulması için SSH önemli bir araçtır.
- Zira, bulutta çalışan sunucular kullanıcılardan uzakta İnternet üzerinde çalışmaktadır. Bunlara erişilmesi için SSH gibi güvenli iletişim yöntemlerine ihtiyaç duyulmaktadır.
- Doğrudan SSH ile uzaktaki sisteme komut penceresi açılabildiği gibi firewall üzerinden uzaktaki sisteme bir tünel oluşturulabilir.

SSH

- SSH, Açık (**`public`**) ve gizli (**`private`**) şifre anahtarı çiftiyle çalışır. Gizli anahtar saklanırken istemcide saklanırken, açık anahtar paylaşılabilir.
- SSH istemcisi, sunucuya bağlandığında, **`login`** şifresiyle giriş yapmasına izin verilmez. Sunucu, istemcinin gizli (**`private key`**) anahtara sahip olup olmadığını sorgular.
  - İstemci bağlandığında anahtar çiftini ifade eden bir ID gönderir.
  - Sunucu bu ID'yi açık anahtarla karşılaştırır. Eğer gönderilenle bu eşleşiyorsa,
  - Tastgele bir sayı üretir ve bunu SSH istemcisine gönderir.
  - SSH istemcisi, bu değeri alarak MD5 hash değerini bulur ve sunucuya gönderir. Sunucu da önceden gönderdiği değeri açık anahtarla MD5 değerini üretir ve istemcinin gönderdiği hash değeriyle karşılaştırır. Eğer gönderilen ve SSH istemcisinden şifreli şekilde alınan değer birbirinin aynıysa, istemcinin şifresiz şekilde sisteme giriş yapmasına izin verilir.



https://www.digitalocean.com/community/tutorials/recommended-security-measures-to-protect-your-servers

SSH nerelerde kullanılır?

- SSH, iki sistem üzerinde çalışan yazılımların birbiriyle haberleşmesi veya bir sistemde yer alan yazılımın diğer bir sistemde kod çalıştırma için de kullanılan güvenilir bir sistemdir.
- SSH varsayılan olarak 22 nolu TCP portundan haberleşir.
- SSH şu işlerde kullanılmaktadır:
  - Bulut sistemlerine bağlantı için
  - Otomasyon sunucusu Jenkins'te uzakta bir düğüm oluşturarak derleme işlerini bu düğümde gerçekleştirmek için
  - Red Hat Ansible'da uzakta yapılacak otomasyon işlerinin merkezden idare edilmesi için
  - Yedekleme yazılımları tarafından
  - vb

GANTEK
INTEGRATING FUTURE

SSH Güvenli mi?

- Açık ve gizli anahtarların korunması çok önemlidir.
- Bazı özellikleri kullanılarak SSH'ın «hack» edilmesi mümkündür.
- Aşağıdaki web sitesi, OpenSSH sunucusunun nasıl «hack» edilebileceğini anlatmaktadır.
  - https://blog.thc.org/infecting-ssh-public-keys-with-backdoors

GANTEK

OpenSSH Sunucusu Kurulumu

- SSH sunucusu kurmak için aşağıdaki komutun terminal penceresinde verilmesi gereklidir.

```
sudo apt install openssh-server
```

- Servisin çalıştığını kontrol etmek gereklidir:

```
sudo systemctl status ssh
```

- Eğer SSH servisi çalışmıyorsa, **`systemctl`** ile aktifleştirilmesi ve ardından başlatılması gereklidir.

```
sudo systemctl enable ssh
sudo systemctl start ssh
```

OpenSSH Sunucusu Kurulumu

GANTEK
INTEGRATING FUTURE

- Eğer **`ufw`** gibi bir firewall kullanılıyorsa, SSH'ın kullanımına izin vermek gereklidir:

```
sudo ufw allow ssh
sudo ufw enable
sudo ufw status
```

- **`ssh`** programı, yapılandırma dosyalarını ev dizininde **`.ssh`** adlı dizin içindeki kullanmaktadır. Bu nedenle bu dizinin **`ssh`** programını kullanmadan önce oluşturulması iyi olur.

```
mkdir ~/.ssh
chmod 0700 ~/.ssh
```

GANTEK

OpenSSH – Kısa Bilgi

- OpenSSH, SSH protokolünün açık kaynak kodlu uygulamasıdır.
- OpenSSH'ın varsayılan değerleri
  - TCP portu : 22
  - Sunucu (daemon) yapılandırma dosyası
    `/etc/ssh/sshd_config`
  - İstemci yapılandırma dosyası:
    `/etc/ssh/ssh_config`
  - Kullanıcı için yapılandırma dosyası
    `~/.ssh/config`
  - Özel ve açık anahtarların (**`id_rsa`** ve **`id_rsa.pub`**) depolandığı yer **`~/.ssh/`**

Yapılandırma dosyaları

- **`/etc/ssh/sshd_config`** dosyası, OpenSSH daemon için parametreleri içeren bir dosyadır.
- **`/etc/ssh/ssh_config`** dosyası, sistemdeki bütün kullanıcılara yönelik çeşitli sunucular için bağlantı parametrelerini içeren bir dosyadır.
- **`~/.ssh/config`** dosyası kullanıcının bağlantı yaptığı sunucularla ilgili bilgileri koyduğu yapılandırma dosyasıdır.

https://www.cyberciti.biz/faq/create-ssh-config-file-on-linux-unix/

ssh-keygen

GANTEK
INTEGRATING FUTURE

- SSH, açık ve gizli anahtar çiftini kullanır. Bu anahtarların oluşturulması için **ssh-keygen** programı kullanılır.
- «**ssh-keygen**» SSH protokolü v2 için anahtar üreten bir programdır.
- Hangi tür anahtar üretileceği, «**-t**» opsiyonu ile belirlenir.
  - Kullanılabilecek değerler olan «**dsa**», «**ecdsa**», «**ecdsa-sk**», «**ed25519**», «**rsa**» anahtar çiftlerinin türünü belirtir. «**rsa**» varsayılan değerdir.
- Açık ve gizli anahtarlara, «**-c**» opsiyonuyla bir yorum (comment) satırının eklenmesi, gelecekte anahtarların ne için üretildiğini ve hangi sistemlere verildiğini anlamamızı sağlayabilir.

```
ssh-keygen -t rsa -C "GC kullanicisi"
```

ssh-keygen

- «**ssh-keygen**» şifreleri oluştururken, anahtarın oluşturulacağı bit sayısını «**-b**» opsiyonuyla belirlememizi sağlamaktadır.

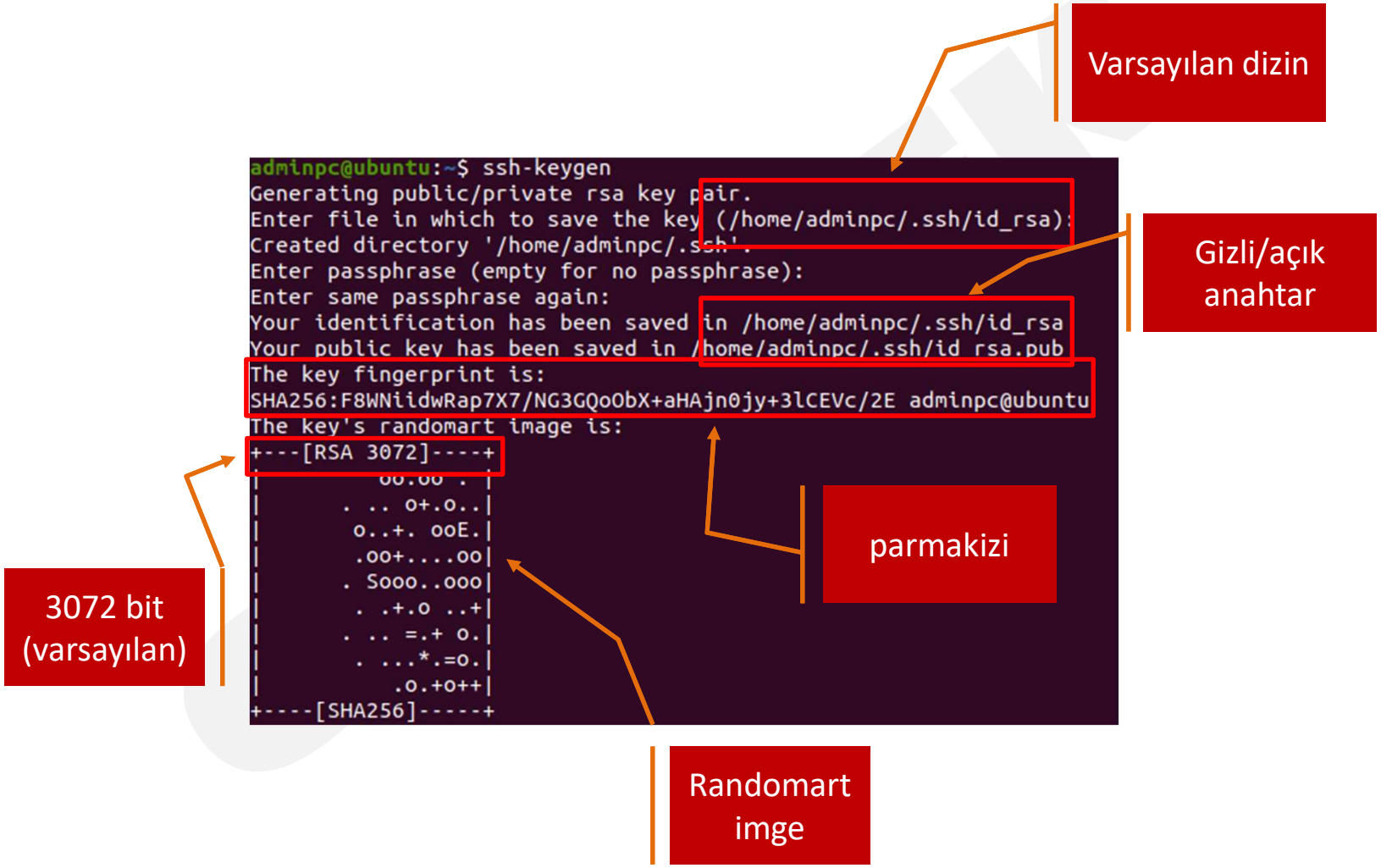
```
ssh-keygen -t rsa -b 8192
```

- Anahtarları üretmek için kullanılacak bit sayısı büyüdükçe iletişim daha güvenli olmakla birlikte istemci ve sunucuya işlem yükü yaratmaktadır.
- Verilebilen minimum bit sayısı 1024 ve «**-b**» parametresi belirtilmezse varsayılan olarak 3072 alınmaktadır.
- Yeterli bir güvenlik seviyesi için (2022) en az 4096 bit tercih edilmelidir.
- Maksimum bit sayısı olarak 16384 verilebilir.

ssh-keygen

- «**ssh-keygen**» çalıştırıldığında, anahtar çiftini nereye koyacağını sorar.
  - Anahtar çiftinin varsayılan olarak tutulduğu yer **~/.ssh/** dizinidir. Bunu değiştirmemeniz iyi olur.
  - Ardından, özel/gizli anahtarınıza bir şifre (**passphrase**) de tanımlayabilirsiniz. Bu güvenliği artırır. Gizli anahtara ancak bu şifreyle erişilebilir.
  - Uzaktan kabuk bağlantısı için güvenliği artırdığı için önerilmekle birlikte, Jenkins, Ansible, vb otomasyon programlarının uzakta kabuk açarak işlem yapmaları için «**passphrase**» girişinin yapılmaması önerilir ve burası «enter» butonuna basılarak geçilmelidir.
- «**ssh-keygen**» **~/.ssh/** dizininde aşağıdaki iki dosyayı oluşturur:
  - **id_rsa.pub** : açık (public) anahtar
  - **id_rsa**: gizli (private) anahtar

ssh-keygen


Varsayılan dizin
Gizli/açık anahtar
parmakizi
3072 bit (varsayılan)
Randomart imge
adminpc@ubuntu:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/adminpc/.ssh/id_rsa):
Created directory '/home/adminpc/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/adminpc/.ssh/id_rsa
Your public key has been saved in /home/adminpc/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:F8WNiidwRap7X7/NG3GQoObX+aHAjn0jy+3lCEVc/2E adminpc@ubuntu
The key's randomart image is:
+---[RSA 3072]----+
| oo.oo . |
| . .. o+.o..|
| o..+. ooE.|
| .oo+....oo|
| . Sooo..ooo|
| . .+.o ..+|
| . .. =.+ o.|
| . ...*.=o.|
| .o.+o++|
+----[SHA256]-----+

ssh-keygen

- «**ssh-keygen**» varsayılan olarak **~/.ssh/** dizini kullanılabilir.
- Bazı durumlarda (yazılım veya kullanıcı için şifre üreteceksenız) farklı bir dizine açık/gizli anahtar çiftinin konulması gerekebilir. Bunun için «**-f**» opsiyonu kullanılabilir. Aşağıdaki komut verildiğinde, **deneme.key.pub** (açık şifre) ve **deneme.key** (özel/gizli şifre) dosyaları oluşturulacaktır.

```
ssh-keygen -t rsa -f deneme.key
```

GANTEK
INTEGRATING FUTURE

ÖRNEKLER: SSH İLE BİR SUNUCU ÜZERİNDE OTURUM AÇMA

https://www.cyberciti.biz/tips/linux-unix-bsd-openssh-server-best-practices.html

`ssh` kullanımı

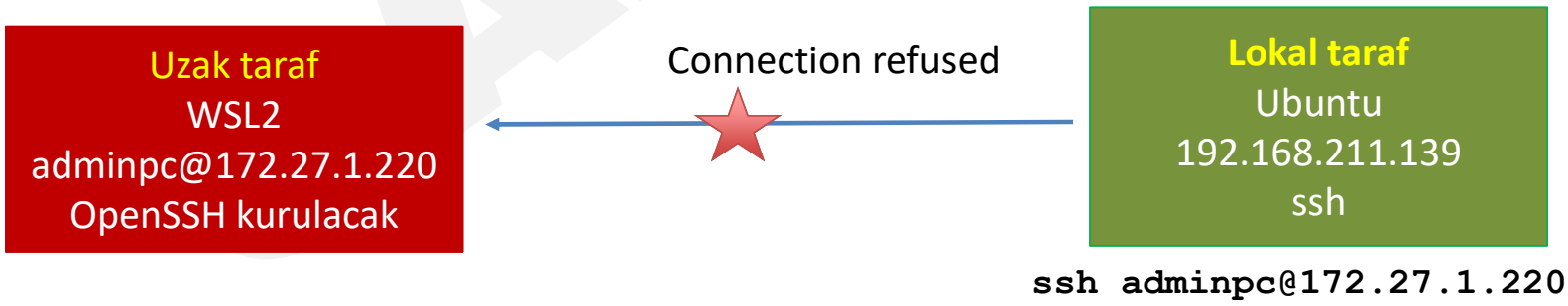
- OpenSSH farklı kimlik onaylama mekanizmaları sunmaktadır.
- **`ssh`** üzerinden iki sistem arasında bağlantı kurulması kolay bir iş olarak görünse de dikkat edilmesi gereken unsurlar bulunmaktadır. En üst seviyede güvenliğin sağlanması için **`ssh`**'ın uygun bir şekilde kullanılması gereklidir.
- Burada vereceğimiz örnekler şunlardır:
  1. Açık/gizli anahtarla ile uğraşmadan **`ssh`** ile doğrudan bir sunucuya bağlanarak, kullanıcı şifresi girerek oturum açmak
  2. Açık/gizli anahtar çifti oluşturarak, kullanıcı şifresi girmeden otomatik olarak oturum açmak
  3. Açık/gizli anahtar çifti oluşturarak, **`scp`** komutuyla güvenli olarak uzaktaki sistemden lokal sisteme dosya kopyalanması

https://www.openssh.com/manual.html

Örnek 1: sunucuya **`ssh`** ile bağlanti

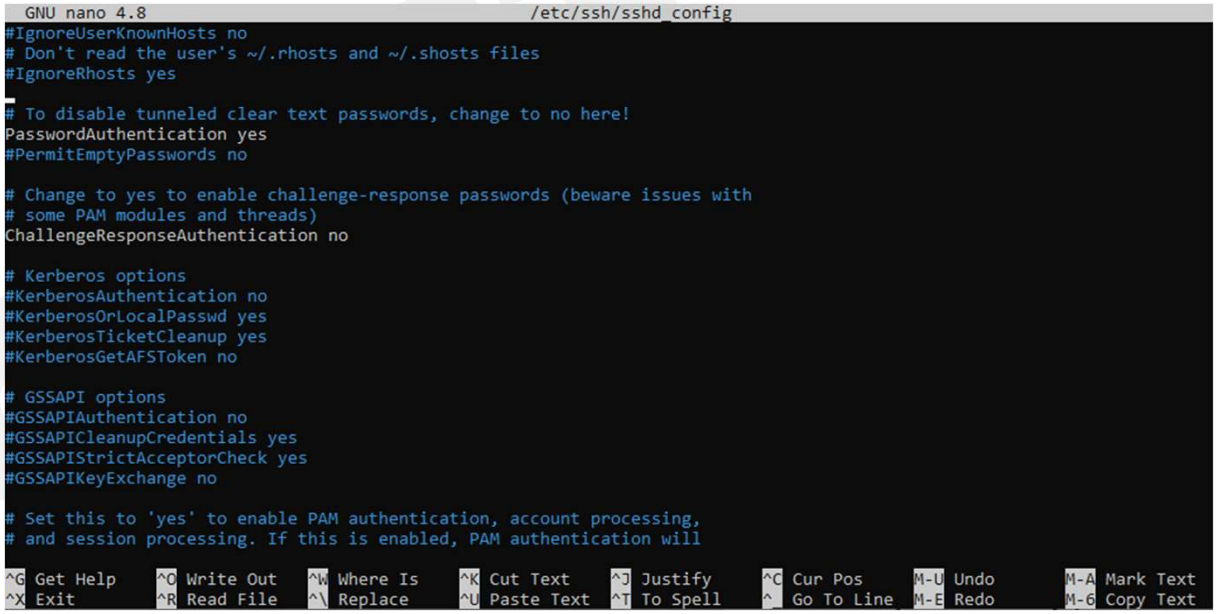
- Lokalde **`ssh`** ile uzak taraftaki «WSL2 Ubuntu» veya «Google Cloud VM» makinesine bağlanmak için, uzak makinede OpenSSH kurulumunun yapılmış olması gereklidir. WSL2'deki kullanıcı isminin **`adminpc`** olduğunu ve IP numarasının **`172.27.1.220`** olduğunu varsayalım.
- Aşağıdaki komutla tünel modunda bağlantı açılabilir.

```
ssh adminpc@172.27.1.220
```



Örnek 1: sunucuya **ssh** ile bağlanti

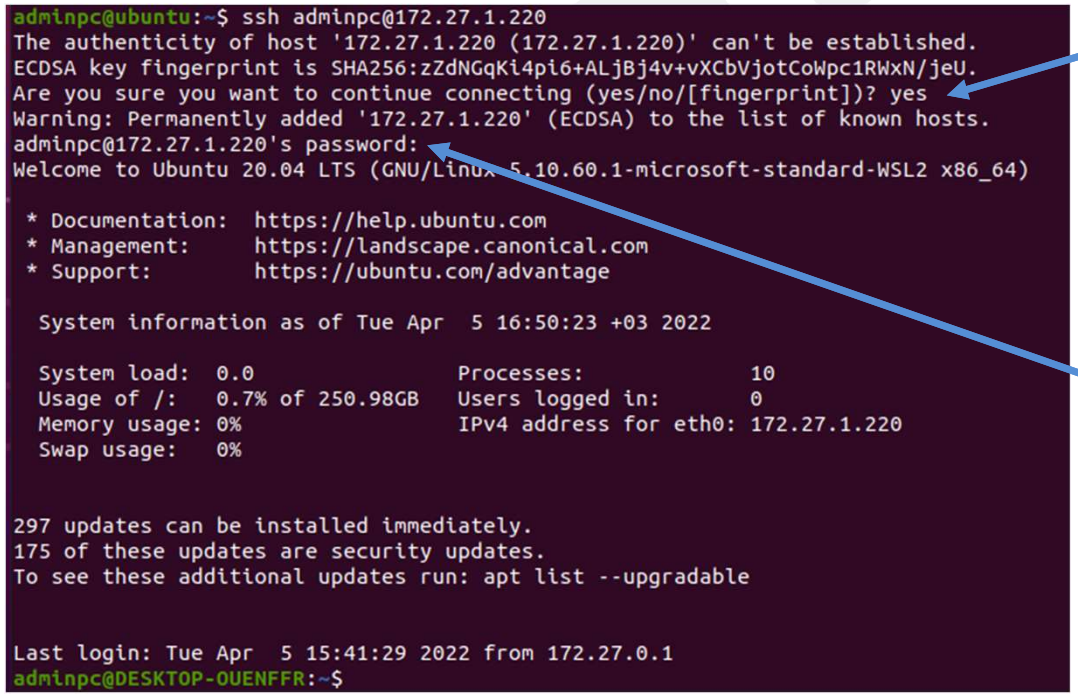
- **ssh**, açık/gizli anahtarların kullanılmadan, kullanıcı şifresi girerek tünel modunda bağlantı kurulabilir
- Kullanıcı şifresine izin vermek için **/etc/ssh/sshd_config** dosyasında «**passwordAuthentication no**» satırı «**passwordAuthentication yes**» yapılmalıdır. Bu şekilde, **ssh** sadece tünel modunda kullanılacak ve kullanıcı adı için şifre yazılması gerekecektir.



man sshd_config

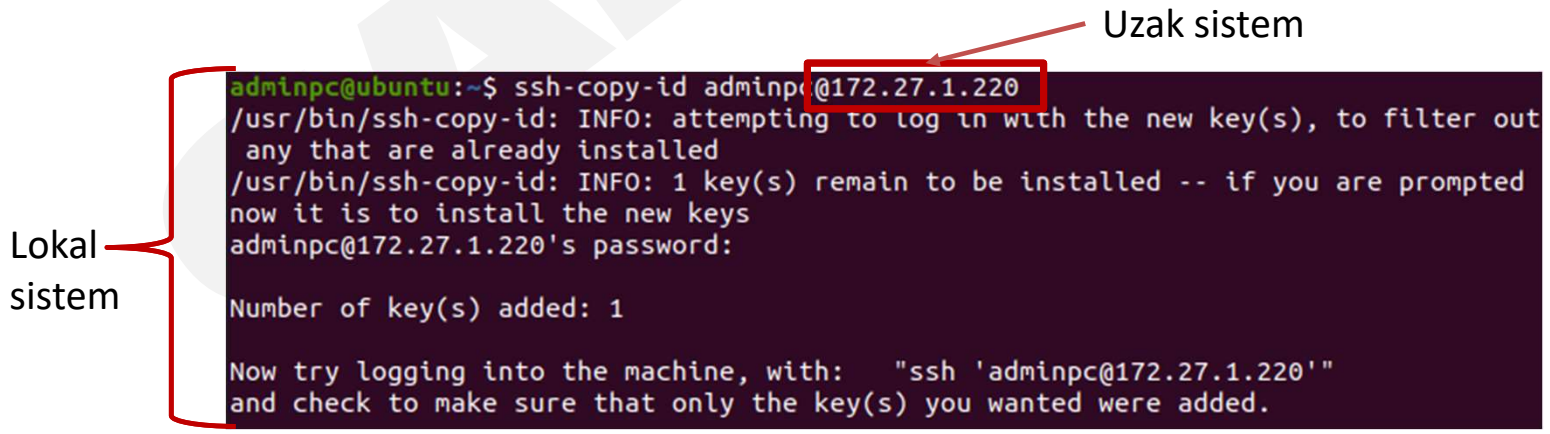
Örnek 1: sunucuya **ssh** ile bağlanti

- **/etc/ssh/sshd_config** yapılan değişiklikten sonra **systemd** kullanan sistemlerde «**sudo systemctl restart ssh**» ve **init** kullanan sistemlerde «**sudo service sshd restart**» komutları verilmelidir. «**ssh adminpc@172.27.1.220**» yazılarak uzak tarafa bağlanılabilir.

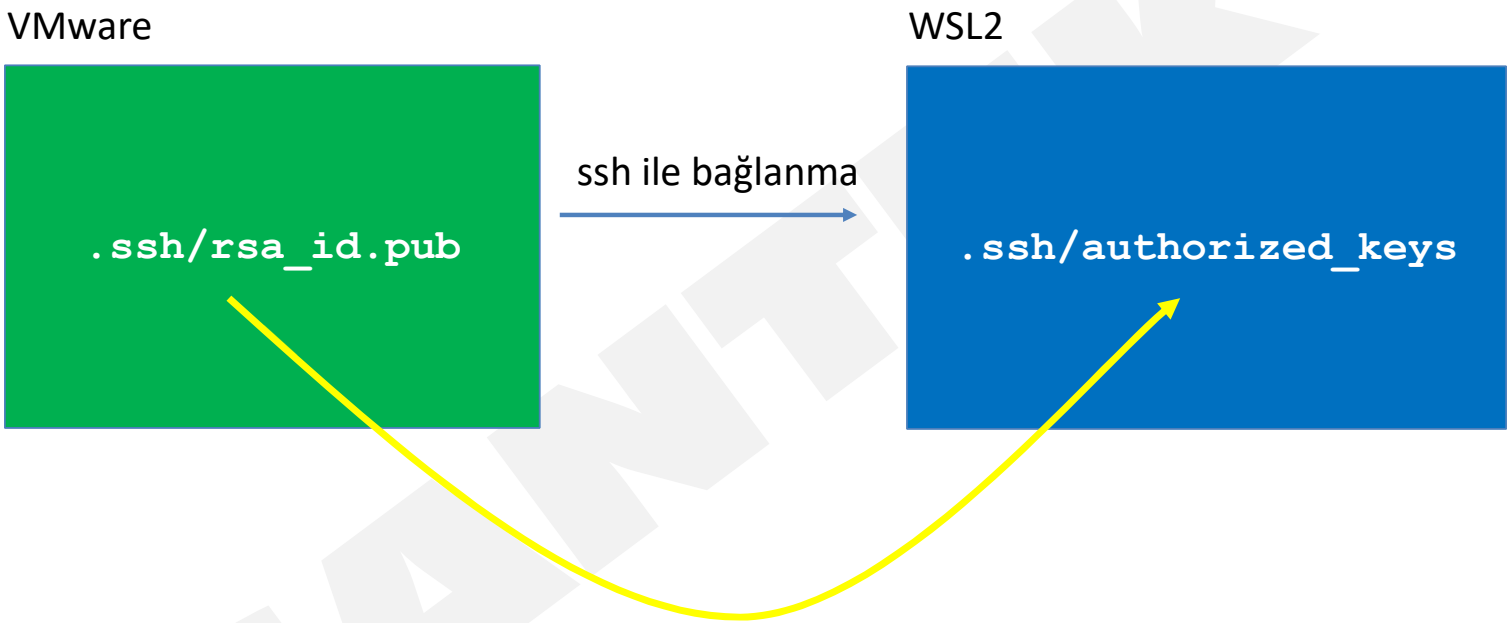


Örnek 2: otomatik giriş

- Bağlanılacak olan sisteme şifre olmaksızın otomatik olarak girilmesi isteniyorsa, lokal tarafta (VMware tarafı) **`ssh-keygen`** programıyla açık/gizli anahtar çiftinin **`~/.ssh/`** dizininde oluşturulması gereklidir.
- Açık/gizli anahtarlar oluşturulduğuna göre, açık anahtarı, bağlanacağımız sisteme (burada WSL2 Ubuntu olacak) **`ssh-copy-id`** komutuyla aktarmamız gerekli. Orada bize belirttiğimiz kullanıcı için bir şifre soracak. Onu girdikten sonra şifre karşı tarafa eklenmiş (karşı tarafta **`.ssh/authorized_keys`** dosyasına) olacak.
- ALTERNATİF: **`ssh-copy-id`** yerine, VMware tarafındaki **`.ssh/id_rsa.pub`** dosyasının içeriğini, WSL2'de **`.ssh/authorized_keys`** dosyasına eklediğimizde de aynı işi yapmış oluruz.



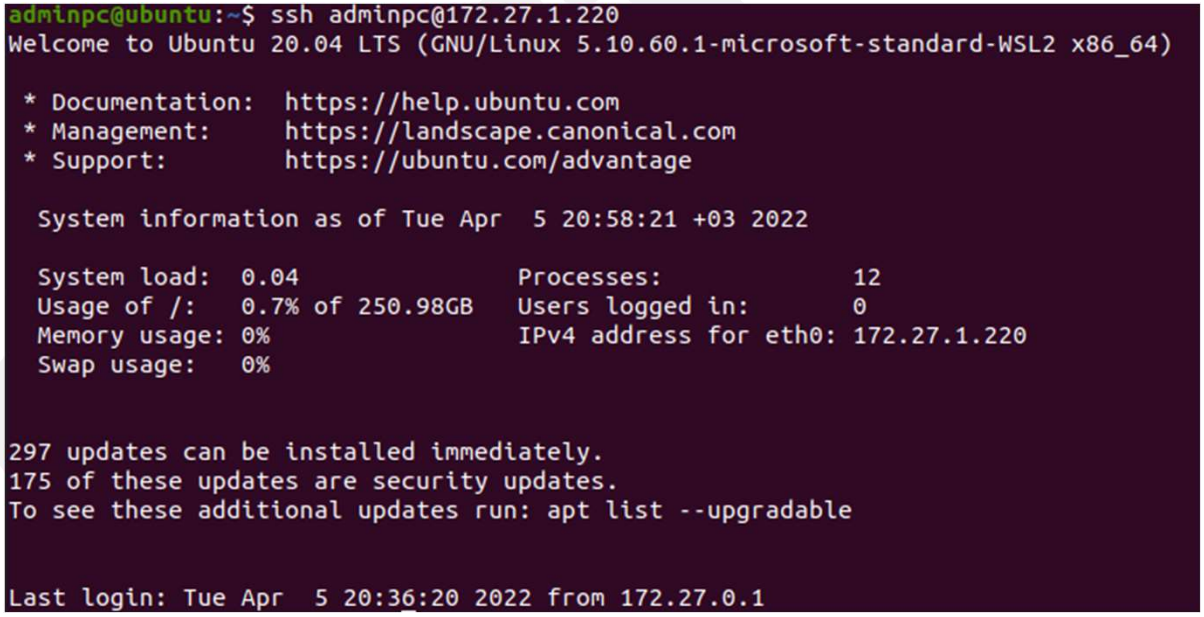
Örnek – 2: şifresiz otomatik giriş



`ssh-copy-id` programıyla ya da VMware'deki açık anahtarı Copy-Paste yaparak uzak sistemdeki **`authorized_keys`** dosyasına ekleyebiliriz.

Örnek 2: otomatik giriş

- **`ssh-copy-id`** komutuyla veya Copy-Paste ile uzak tarafa aktardığımız ve şifre girdiğimiz için kabul edilmiş olan açık anahtarımız, tünelin oluşturulmasında kullanılacak ve giriş için ayrıca şifre girilmesine gerek olmayacak.
- «**`ssh adminpc@172.27.1.220`**» denilerek uzakta tarafa otomatik olarak girilecektir.



Örnek 3: güvenli dosya kopyalama

- **`scp`** komutu, uzaktaki sistemden güvenli bir şekilde herhangi bir dosya kopyalanmasını sağlayan OpenSSH komutudur (Örnek 2'yi uyguladıktan sonra kullanılabilirsiniz)
- Uzak taraftaki sistemde (Google Cloud veya WSL2'de) **`sil.txt`** isimli bir dosyanın **`adminpc`** kullanıcısının ev dizininde bulunduğunu varsayalım.
- Uzaktaki **`sil.txt`**'nin lokal bilgisayara kopyalanması için şu komut kullanılabilir:

```
scp adminpc@172.27.1.220:sil.txt .
```

```
adminpc@ubuntu:~$ scp adminpc@172.27.1.220:sil.txt .
sil.txt                                       100%    5     1.1KB/s   00:00
```

scp (secure copy)

- **scp** (secure copy) : Uzaktaki bilgisayardan bir dosyanın lokal dosya dizinine güvenli bir şekilde aktarılmasını sağlayan bir programdır.
- Uzaktaki **foobar.txt** dosyasını **/lokal/dizin**'e aktarır.

```
scp hgsenel@hgsenel.org:foobar.txt /lokal/dizin
```

- Lokaldeki **foo** dizinini uzak makinedeki **/uzak/dizin**'e aktarır.

```
scp -r foo user1@35.11.11.11:/uzak/dizin
```

http://www.hypexr.org/linux_scp_help.php

GANTEK

Örnek 4: şifreyle girişi engelleme

- Güvenliğin artırılması için, Örnek 1'de yapılabildiği gibi kullanıcının şifreyle oturum açmasının engellenmesi gerekebilir. Ayrıca, boş şifrelerin de engellenmesi iyi olacaktır.
- Uzak sistemde SSH daemon'unun yapılandırma dosyası olan **`/etc/ssh/sshd_config`**'e aşağıdaki satırlar eklenmeli ve varsa daha önce eklenmiş satırlar silinmelidir. Ardından **`sshd`** servisi yeniden başlatılmalıdır.

```
AuthenticationMethods publickey
PubkeyAuthentication yes
passwordAuthentication no
PermitEmptyPasswords no
```

Örnek 5: Belirli kullanıcılara izin verme

- Güvenlik seviyesinin artırılması için uzaktaki sistemde SSH üzerinden oturum açma izni sadece belirli kullanıcılara verilmelidir.
- Bu amaçla, **`/etc/ssh/sshd_config`** dosyasına «**`AllowUsers`**» yanına izin verilen kullanıcıların login isimlerinin yazılması gereklidir:

```
AllowUsers adminpc
```

- Ardından **`ssh`** servisi yeniden başlatılmalıdır.

Örnek 6: SSH portunun değiştirilmesi

- Varsayılan olarak SSH portu 22'dir. SSH zafiyetleri için ilk bakılan portun 22 olması nedeniyle, bunun farklı bir porta taşınması, güvenlik açısından iyi olabilir.
- Ayrıca, varsayılan olarak SSH bütün IP'lerden gelen taleplere açıktır. Bunun kısıtlanması ve belirli IP'lere izin verilmesi için **`sshd_config`** dosyasına şu satırlar eklenebilir:

```
Port 450
ListenAddress 192.168.211.67
ListenAddress 171.45.33.22
```

- «**`man sshd_config`**» ile ayrıntılara bakılabilir.

Örnek 7: zaman aşımının ayarlanması

- **`ssh`** üzerinden bağlanan bir kullanıcı, bağlı kaldığı sürece oturumu açık kalır.
- Bu durum, bazı güvenlik risklerini de beraberinde getirmektedir.
- Bu nedenle belirli bir süre (örneğin 600 saniye) pasif durumda kalan kullanıcının sistemden çıkarılması iyi olacaktır.
- Bu amaçla **`/etc/ssh/sshd_config`** dosyasına aşağıdaki satırların eklenmelidir:

```
ClientAliveInterval 600
ClientAliveCountMax 0
```

- Değişikliklerin devreye girmesi için **`ssh`** servisinin yeniden başlatılması gereklidir.

Örnek 8: kullanıcıya **sudo** yetkisi

- Yeni oluşturulan bir kullanıcı **ssh**'la kendi oturumunu açabilir.
- Ancak, bu kullanıcı **ssh**'la sisteme bağlandığında **sudo** kullanma yetkisi olmayabilir.
- Bu kullanıcıya **sudo** yapma yetkisi, süper kullanıcı (root) tarafından şu komutla sağlanabilir:

```
sudo adduser <kullanıcı-adı> sudo
```

- Not: kullanıcı **login** durumundaysa oturumu kapatıp tekrar giriş yapması gerekecektir.

«**hash**» ne demek?

- Kriptografik «**hash**» algoritmaları, uzun bir veriyi alarak onu temsil eden ama daha kısa bir metin oluşturmak için kullanılır.
- Oluşturulan «**hash**» değerinden «**hash**» yapılan uzun metni türetmek imkansızdır.
- Dosya uzunluğu ne olursa olsun, «**hash**» değeri aynıdır (**hash** uzunluğu algoritmaya göre değişir)

https://www.flaticon.com/free-icon/google-docs_2991112

«**hash**» ne demek?

GANTEK
INTEGRATING FUTURE

- Ne işe yarar?
- «**Hashing**» algoritmaları şu işlere yarar:
  - bir dosyanın değişip değişmediğini anlamamıza yarar. Dosyanın 256 bitlik «**hash**» değeri bulunur ve bir yere konur. Eğer dosya değişirse, dosyanın o haliyle oluşturulacak «**hash**» değeri de değişeceğinden dosyanın farklı bir dosya olduğu anlaşılır.
  - Şifre yerine Linux sistemde kullanıcı şifreleri tutulmaz. Sadece hash değerleri tutulur. Kullanıcı sisteme girerken verdiği şifre de «**hash**»lenir ve şifrenin «**hash**»iyle yeni oluşturulan «**hash**» değeri karşılaştırılarak eğer uyumluysa sisteme giriş yaptırılır.
- SHA256, yaygın olarak kullanılan bir algoritmadır. Herhangi uzun bir metinden veya dosyadan 256 bitlik «**hash**» veya «**message digest**» değeri üretilir.
- MD5, 1991'de geliştirilmiştir ve 16 baytlık **hash** üretmektedir (güvenilir değildir)

«key fingerprint» ve «randomart»

- SSH istemcisi bir sunucuya ilk defa bağlandığında, sunucunun açık (public) anahtarının parmak izi (**fingerprint**) kullanıcıya gösterilir ve sunucunun gerçek olduğunu onaylaması istenir.

```
$ ssh anoncvs.de.openbsd.org
The authenticity of host 'anoncvs.de.openbsd.org (131.188.40.91)'
can't be eastablished.
RSA key fingerprint is fc:94:b0:c1:e5:b0:98:7c:58:43:99:76:97:ee:9f:b7
Are you sure you want to continue connecting (yes/no)?
```

- Parmak izinin ne tür bir değer olduğunu hatırlamak ve karşılaştırma yaparak «bu sunucu benim bildiğim sunucu» demek neredeyse imkansızdır.
- Onaltılık (hex) değerler yerine, bunu temsil eden ASCII karakterlerden oluşturulmuş bir resmin gösterilmesinin daha uygun olacağı düşünülmüş ve OpenSSH v5.1'de «parmak izi görselleştirme» özelliği eklenmiştir.

http://www.dirk-loss.de/sshvis/drunken_bishop.pdf

Gizli (private) anahtar

GANTEK
INTEGRATING FUTURE

```
adminpc@DESKTOP-OUENFFR:~$ more .ssh/id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAr6vm0o1dFkmhWXu8H9azsL/BIiG7WC+94SVJhiSOj3Qcygj+ecgW
IEKDlDNb8VXEvtSz7oQ6aOcVGyRqLEfs3pI+BfcK4/GMH5ptx+W8Dtju5Ug+5wv4vpr+C0
VpBU1QhuzYBy2URMzxEg/O7NsBafMnv3aMsC8wmBGRZYiEEmfsAq1j0ENedFKofaXfVon1
Ba7B9Bkq/ksEvfw+Vxq3DGjmDqjUkHWMy0RxeYX/Vj89f6fBP7vorbj76qSkTO47qpErs3
wCehrs81z6K1sbLaNJm1YYpOT16nWOEuTqR5wGSRnQk5jMkDcezi591E0winJ3iuh+15sU
yftylp5GTwHsZofEVfsvDdluSrZ9fLvBk2xFyg1Hx/Avh62tIFhr2Wsf2MVEDPtCxfOjrN
hLNwm8hKpW1/QctuBEcNWgEXCJUnt258e4AcSr3Q50EuwiOv9xUcCtkkRmvYVmHuZb13JU
6hd7x2q01iYScqvH3CcwfoBBJelPA9yutiIIRll7AAAFkIGK3l2Bit5dAAAAB3NzaC1yc2
EAAAGBAK+r5tKNXRZJoVl7vB/Ws7C/wSIhu1gvveElSYYkjo90HMoI/nnIFiBCg5QzW/FV
xL7Us+6EOmjnFRskaixH7N6SPgX3CuPxjB+abcflvA7Y7uVIPucL+L6a/gtFaQVNUIbs2A
ctlETM8RIPzuzbAWnzJ792jLAvMJgRkWWIhBJn7AKtY9BDXnRSqH2l31aJ9QWuwfQZKv5L
BL38Plcatwxo5g6o1JB1jMtEcXmF/1Y/PX+nwT+76K24++qkpEzuO6qRK7N8Anoa7PJc+i
tbGy2jSZtWGKTk5ep1jhLk6kecBkkZ0JOYzJA3Hs4ufdRNMIpyd4rofpebFMn7cpaeRk8B
7GaHxFX7Lw3Zbkq2fXy7wZNsRcoNR8fwL4etrSBYa9lrH9jFRAz7QsXzo6zYSzcJvISqVt
f0HLbgRHDVoBFwiVJ7dufHuAHEq90OdBLsIjr/cVHArZJEZr2FZh7mW9dyVOoXe8dqtJYm
EnKrx9wnMH6AQSXpTwPcrrYiCEZZewAAAAMBAAEAAAGBAJ2GSPn/FzzYE54ZDU2DiifxGu
9PTh462Uqd0/s8prVOSk6EmCR93hFZc4GeOFEFNuJj+2DjIzlUFaH8GlTQvj+q5YEV0Kcr
vxnLsatxfxvVPQtOqFEodAn/E9Tpjc3aUV5b3S86mNzE5Vrr4AKWNANcrgTK+kc2WZ3TCw
z/yGZ017Zi4sEq1/T49euNpOKF3LgWyAbp1pqmjH8MRJ547zFqkvkMPWe4cOMI4bQ2Cwu8
s5nXxLsw9elpmkQDFO2DJXJ8cy8Z6wEY6x1qsvYcSfdbDO8O+4L72VoBJj46dOsOUXBazI
BcndIAVN6JBgDW3sBXrbbyWZieMgwKvw2RpZBLLmatScs2rq4ocxir7JX3dG2n7PQ6NYRj
gjFXOfa2YjxZy8Lp7JqSnB0oPLqC/0DPpQ8/3BQsK2uGJolv+qITJScbDsSnUJ77RQxN+A
wxH20rXP0MXQphVZWHd8s3Sh8eNnEYmUpDt3m8cLoc16z19TJ1nl91wykNTQVHnIRGYQAA
AMEA1bcoWosKu+wdoxwdx8EO3Aforn9+NHZyZrRcZ0ybztTJWfh+1yxmjmLJhH9skddYtq
udwZsepeYhZRGJL7NYxUIQLoSTe//JWyq/A/+IOYsOueERdM66sw1urKswOHh637PtlYUp
LGx6ThfIH3iMNvf9i+OTxR3OOnarPD3S9dv2B6d81FHIj6CwbMag+3Tft5i3akHt/6aFV8
mmISY5gascJkTAMfIRBfdjO161VymGNlABCyI55dHNqKG3xSHLAAAAwQDjl+muYpu94wiG
cRAN2/zJuf9GlYHsFR7QnRhYLUfpD17REy2lKtuVkmbx6tiQR0RG2/tANM0K7tX61Q61df
0ZeH6cmwqWg7MpDat0t0GWFj9ZFcvlfFcJVtybc3LQ2tycYc1tWChB0x3KiAsGsPzQAFvP
MZ+Fukgk/7W1Cte0gxw6pGoV4bzPTghHv4uMDmFbcJbwm4URp0UdLKHFOQp0wZJTGk6kTo
KMV3ZdI+YvgoxkWy01Q1i7gwxNee9+/EsAAADBAMWY+XuPl8gN7rxNfc7llIpR2KWukJI1
u5nsiDhAN6zmFaNi6OmtJPnm97fj+zs2DCecS+IVXrVCpOZUg1aplfGQYxPYn8Pzl5htVL
ciInc18wxbUUv/rh0yn9GeN+W/M//PYTx9WtoAc8cfbKynd15wHnXgPUuqFjFYkNx2FcSR
AjekxYKpz/lHvpkJfJEW5kdQiGKy4uecHdQPDfeEynSJS4Q2sHoNBiS2E/fia+NpI6HMGo
AG953NrbTLsA55kQAAABdhZG1pbnBjQERFU0tUT1AtT1VFTkZGUgEC
-----END OPENSSH PRIVATE KEY-----
```

Açık (public) anahtar

- Açık anahtarın ilk boşluğa kadar olan kısmında «**ssh-rsa**», «**ssh-ed25519**» gibi değerler olabilir. Açık anahtar, **base64**'le kodlanmıştır.

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQCvq+bSjV0WSaFZe
7wf1rOwv8EiIbtYL73hJUmGJI6PdBzKCP55yBYgQoOUM1
vxVcS+1LPuhDpo5xUbJGosR+zekj4F9wrj8Ywfmm3H5bw
O2O7lSD7nC/i+mv4LRWkFTVCG7NgHLZREzPESD87s2wFp
8ye/doywLzCYEZFliIQSZ+wCrWPQQ150Uqh9pd9WifUFr
sH0GSr+SwS9/D5XGrcMaOYOqNSQdYzLRHF5hf9WPz1/p8
E/u+ituPvqpKRM7juqkSuzfAJ6GuzyXPorWxsto0mbVhi
k5OXqdY4S5OpHnAZJGdCTmMyQNx7OLn3UTTCKcneK6H6X
mxTJ+3KWnkZPAexmh8RV+y8N2W5Ktn18u8GTbEXKDUfH8
C+Hra0gWGvZax/YxUQM+0LF86Os2Es3CbyEqlbX9By24E
Rw1aARcIlSe3bnx7gBxKvdDnQS7CI6/3FRwK2SRGa9hWY
e5lvXclTqF3vHarSWJhJyq8fcJzB+gEEl6U8D3K62IghG
WXsW adminpc@DESKTOP-WIN
```

Konak Anahtarı (Host Key)

- Her konak (host) özgün bir konak anahtar (host-key) çiftine gerek duyar. Konak anahtarlarının paylaşımı tavsiye edilmemekle birlikte, paylaşıldığında bazen pratik gereksinimleri karşılayabilir.
- OpenSSH'ta, konak-anahtar ilk kurulduğu zaman veya ilk boot ettiği anda oluşturularak **`/etc/ssh`** dizininde **`ssh_host_<anahtar-algoritması>_key`** dosyalarına konulur. Her bir anahtar algoritması için bir tane dosya oluşturulur.

```
adminpc@deneme:~$ ls /etc/ssh
moduli                 ssh_host_dsa_key       ssh_host_ed25519_key      ssh_import_id
ssh_config             ssh_host_dsa_key.pub   ssh_host_ed25519_key.pub  sshd_config
ssh_config.d           ssh_host_ecdsa_key     ssh_host_rsa_key          sshd_config.d
ssh_config.dpkg-dist   ssh_host_ecdsa_key.pub ssh_host_rsa_key.pub      sshd_config.ucf-dist
adminpc@deneme:~$
```

https://www.ssh.com/academy/ssh/host-key

Konak Anahtarı

- **`ssh`** istemcileri, bir sunucuya bağlantı kurdukları ilk anda, karşı tarafın açık konak-anahtarını alırlar ve **`/etc/ssh/known_hosts`** ve **`~/.ssh/known_hosts`** dosyalarına eklerler. Bir dahaki bağlantıda bunu kontrol ederek ilerlerler.
- Konak anahtarları, çiftler halinde oluşturulurlar ve gizli anahtara sadece süper kullanıcı (root) erişebilir.
- Gizli anahtarın bir saldırgan tarafından elde edilmesi önemli bir güvenlik riskidir. İstemciler bağlandığında araya istenmeyen komutlar ekleyebilir ve çalıştırabilir.
- Büyük kuruluşlarda konak anahtarların yönetimi için çaba sarf edilmelidir. Örnek bir sistem:

`https://www.ssh.com/academy/iam/ssh-key-management`
`https://www.youtube.com/watch?v=ke_M3t_L3iY`

https://www.ssh.com/academy/ssh/host-key

~/.ssh/config

- İstemci tarafında, girişleri hızlandırmak için **~/.ssh/config** dosyasına bazı parametreler eklenebilir.

```
### butun hostlar icin ##
Host *
     ForwardAgent no
     ForwardX11 no
     ForwardX11Trusted yes
     User adminpc
     Port 22
     Protocol 2
     ServerAliveInterval 60
     ServerAliveCountMax 5

## override as per host ##
Host gcubuntu
     HostName 172.27.1.220
     User adminpc
     Port 22
     IdentityFile /home/adminpc/.ssh/id_rsa
```

Bu tanımdan sonra komut satırından «**ssh gcubuntu**» şeklinde girilebilir.

https://www.cyberciti.biz/faq/create-ssh-config-file-on-linux-unix/

~/.ssh/config

GANTEK
INTEGRATING FUTURE

- **Host**: Yeni bir konak tanımı bu kelimeden sonra başlar. Kelimeden sonraki * bütün konaklar için ortak tanımları simgeler. Eğer * yoksa, SSH sunucusunun ismi verilir. Bu isim, komut satırından girişlerde kolaylık sağlar ve doğrudan «**ssh host-ismi**» denilerek gerçekleştirilebilir.
- **Hostname**: hostun ismi veya IP numarası
- **User**: girişte kullanılacak kullanıcı ismi
- **IdentityFile**: ssh istemcisi için kullanılacak açık anahtar. Her sunucu için farklı bir açık/gizli anahtar çifti oluşturulabilir. Girilmezse **~/.ssh/id_rsa** anahtarı varsayılan olarak kullanılabilir.
- **ProxyCommand**: bağlantıda Proxy kullanılacaksa Proxy komutu
- **LocalForward**: lokal makinedeki TCP portunun güvenli kanal üzerinden belirtilen sunucuya gönderileceği ve oradan çıkış yapacağı belirtilir.
- **Port**: Hangi porttan sunucuya bağlanılacağı (varsayılan 22)
- **Protocol**: 1: ssh v1, 2: ssh v.2
- **ServerAliveInterval**: karşı taraftan ne kadar sürede veri alınmazsa bağlantının koparılacağını belirtir.
- **ServerAliveCountMax**: .sunucuya gönderilen «hayatta mısın?» mesajlarının kaç tane olacağını söyler.

https://www.cyberciti.biz/faq/create-ssh-config-file-on-linux-unix/

ssh-audit

- **ssh** sunucu ve istemci sistemlerinin izlenmesi ve risk yaratabilecek özelliklerin görülmesi güvenlik için önemlidir.
- **ssh** sisteminin izlenmesi için «**ssh-audit**» paketi kullanılabilir.

```
sudo apt install ssh-audit
```

```
adminpc@DESKTOP-OUENFFR:~$ ssh-audit localhost
# general
(gen) banner: SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.4
(gen) software: OpenSSH 8.2p1
(gen) compatibility: OpenSSH 7.3+, Dropbear SSH 2016.73+
(gen) compression: enabled (zlib@openssh.com)

# key exchange algorithms
(kex) curve25519-sha256                     -- [warn] unknown algorithm
(kex) curve25519-sha256@libssh.org          -- [info] available since OpenSSH 6.5, Dropbear SSH 2013.62
(kex) ecdh-sha2-nistp256                    -- [fail] using weak elliptic curves
                                            `- [info] available since OpenSSH 5.7, Dropbear SSH 2013.62
(kex) ecdh-sha2-nistp384                    -- [fail] using weak elliptic curves
                                            `- [info] available since OpenSSH 5.7, Dropbear SSH 2013.62
(kex) ecdh-sha2-nistp521                    -- [fail] using weak elliptic curves
                                            `- [info] available since OpenSSH 5.7, Dropbear SSH 2013.62
(kex) diffie-hellman-group-exchange-sha256  -- [warn] using custom size modulus (possibly weak)
                                            `- [info] available since OpenSSH 4.4
(kex) diffie-hellman-group16-sha512         -- [info] available since OpenSSH 7.3, Dropbear SSH 2016.73
(kex) diffie-hellman-group18-sha512         -- [info] available since OpenSSH 7.3
(kex) diffie-hellman-group14-sha256         -- [info] available since OpenSSH 7.3, Dropbear SSH 2016.73
```

https://www.cyberciti.biz/tips/how-to-audit-ssh-server-and-client-config-on-linux-unix.html

ssh-audit

- **ssh-audit**'in kontrol ettiği algoritma grupları şunlardır:
  - **Ciphers**: izin verilen şifreleme algoritmaları, virgülle ayrılarak yazılabilir. **ssh-audit** tarafından zayıf olduğu bildirilenler yazılmaz.
  - **MACs**: (mesaj doğrulama kodu): veri öbeğinin değiştirilmediğini kanıtlayabilmek için kullanılan veriye eklenen özet niteliğinde küçük boyutlu doğrulama kodunu oluşturma algoritmalarıdır.
  - **Kex**: (Key Exchange algoritması): bağlantı kuran sistemlerin açık anahtarlarını birbirilerine güvenli şekilde aktardıkları algoritmadır. Diffie-Hellman algoritması, herhangi bir zayıflığı bulunmaması sebebiyle yaygın olarak kullanılan algoritmadır.
  - **Key**: Konak anahtarı algoritmaları: **ssh** sunucusunun istemciye kendisini doğrulaması için kullandığı açık anahtar temelli algoritmalardır.

https://www.openssh.com/legacy.html

GANTEK
INTEGRATING FUTURE

ssh-audit

- **ssh-audit** tarafından gösterilen zayıf algoritmaların kullanılmaması için **/etc/ssh/sshd_config** dosyasında değişiklikler yapılmalıdır.

GANTEK

http://kb.ictbanking.net/article.php?id=690
http://kb.ictbanking.net/article.php?id=691&oid=1

GANTEK
INTEGRATING FUTURE

IPSEC

GANTEK

IPSec nedir?

GANTEK
INTEGRATING FUTURE

- İnternet Protocol (IP) katmanında, bilgisayarlar arasındaki IP paketlerinin şifrelenmesiyle güvenliği sağlamak için kullanılan protokollerdir.
- Yukarıdaki katmanlarda (örneğin TCP, Uygulama katmanlarında) verinin şifrelenmesi için farklı yöntemler ve algoritmalar kullanılır. TLS, TCP katmanında çalışırken, SSH ise uygulama katmanındadır. **IPSec** IP katmanındadır.
- Bu anlamda **IPSec** ile yapılan iletişimde, yukarıdaki katmanlarda hangi uygulamaların olduğu ve hangi uygulama paketlerinin geçtiği önemli değildir. Aynı **IPSec** bağlantısı üzerinden farklı uygulamaların paketleri (**web**, **http**, **ftp**, vb) güvenli şekilde aktarılabilir.
- **IPSec**, **VPN** (Virtual Private Network) uygulamalarında kullanılan protokollerden biridir.

IPSec

- **IPSec** 1995-2005 arasında geliştirilmiştir ve IPv6'nın geliştirilmesi sırasında temel alınan özelliklerden biridir.
- IPv4 yığıtlarında **IPSec** opsiyoneldir ama IPv6 içine gömülü şekilde gelerek IP paketlerinin şifrelenmesini sağlar. IPv4'te IPSec özelliği bulunan bazı IP yığıtları da bulunmaktadır.
- **IPSec**, IPv4 için ek bir özellik olmasına rağmen, IP paketlerinin şifrelenmesi için en fazla kullanılan yöntemdir.

IPSec

- **IPSec**'in iki ana işlevi vardır:
  - Şifreleme (encryption)
  - Kimlik doğrulama (authentication)
- **IPSec** iki modda çalışır:
  - Taşıma (transport)
  - Tünelleme (tunneling)
- Taşıma modunda, **IPSec** iki bilgisayar arasındaki trafiği şifreler. IP başlıkları şifrelenmezken, içindeki veriler şifrelenir.
- Tünellemede, **IPSec**, iki alt ağ arasında sanal bir tünel oluşturur ve bu ağlar arasında akan trafik şifrelenir. IP paketlerindeki başlıklarla birlikte veri bölümü de şifrelenir.
- Normal şartlarda, iki konak arasında akan IP paketleri için bir bağlantı oluşturulmasına gerek duyulmaz. TCP paketleri için bir bağlantı sağlanması gereklidir.

IPSec - Avantajları

- Uygulamadan bağımsız şekilde, IP paketlerinin şifrelenerek aktarılmasını sağlar.
- IP paketlerinin şifrelenerek gönderilmesi, gönderildiği bilgisayarın üzerinde hangi sistemlerin bulunduğunun açığa çıkmasını engeller. (Sadece bir TCP portundan yapılan iletişimin şifrelendiğini ve diğer IP paketlerinin açık olduğunu düşünün)
- **IPSec** temelli VPN uygulamalarında, hangi tür paketlerin aktarıldığı önemli değildir. IP başlık ve veri bölümlerinin tamamı şifrelenir ve güvenlik sağlanır.

IPSec - Dezavantajları

GANTEK
INTEGRATING FUTURE

- **IPSec** temelli VPN uygulamaları, uzaktaki bilgisayarın ağın içine alınmasını sağlar. Burada yer alabilecek kötücül yazılımın tüm ağa sirayet etmesi mümkündür. (Çünkü **IPSec** uygulamalardan gelen paketleri dikkate almaz).
- **IPSec** kapsamlı bir standarda sahiptir ve bütün bu özelliklerin hepsinin **IPSec** kullanan uygulamalarında kullandığı söylenemez. Bu durum bağlanılan tarafla ilgili uyum sorunları yaratabilir.
- **IPSec**'in sistemlerde yoğunluk yarattığı bilinmektedir.
- **IPSec**'i geliştiren kişilerin FBI/NSA gibi kurumlarla çalışarak arka kapılar bıraktıkları ve bunlar kullanılarak VPN iletişimlerinin izlendiğine dair bazı iddialar dile getirilmektedir. Ancak bunlar iddia niteliğindedir ve henüz bunu doğrular kanıtlar bulunamamıştır.