PKI ve SSL

Bu bölümde PKI'da *(Public Key Infrastructure)* kullan lan, Sertifikalar, Anahtarlar ve Otoriteler dahil olmak üzere, anahtar terimlere genel bir bakış sunulmaktadır. Ayr›ca PKI taraf›ndan kullan lan hizmetler ve kamusal anahtar kriptolamas ile ilgili teknikler üzerinde durulmaktadır. PKI birden fazla şeyi ifade etmek için kullanıld - ğından biraz kafa karıştırıcı olabilir. PKI bir yandan bir güvenlik altyapısını sağla- mak için bir arada kullan lan yöntemler ve teknolojileri ifade ederken öte yandan içeriğin doğrulanması için kullanılan kamusal ve özel anahtar çiftinin kullanımını da ifade edebilir. PKI'ın kullanıcılarına aşağıdaki faydaları sağlaması beklenir:

- Elektronik olarak alınıp verilen bilginin niteliğinin kesinliği.
- Bilginin kaynak ve alıcısının kesinliği.
- Bilginin zaman› ve zamanlamas›n n garantisi.
- Bilginin gizliliğinin kesinliği.
- Bilginin mahkemede delil olarak kullanılabileceğinin garantisi.

Bu hizmetler kamusal anahtar kriptografisi olarak anılan, gönderenin kimliğinin doğrulanmas *(İmzalama)* ve/veya gizliliği sağlamak *(Kriptolama)* için ilgili kriptografik anahtar çiftini kullanan matematiksel bir teknik ile sağlanır.

1. Kamusal Anahtar Kriptografisinin Çalışması

Kamusal anahtar kriptografisi, matematiksel olarak birbiri ile ilişkilendirilmiş kriptografik bir anahtar çiftini kullanır. Eğer anahtardan biri bilgiyi kriptolamak için kullanılırsa, kriptolanmış bilgi ancak diğer anahtar ile çözülebilir. Anahtarlardan birine sahip olmak, diğerini üretebileceği anlamına gelmez; kullanılan algoritmadan dolayı, imkansız olmasa da diğer anahtarı üretmek çok zordur. Sonuç olarak bir Kamusal Anahtar Sisteminde temel bileşenler:

- **Bir kamusal anahtar.** Bu serbest olarak dağıtılan ve ağ üzerinde tüm kullanıc - ların içeriğini görebileceği anahtard r.
- **Karşılık gelen bir gizli anahtar.** Gizli anahtar diğer ağ kullanıcılarına açık ol- mayan, kimlik ispat› için kullan›lan hususi anahtard r.

Anahtarlar›n üretilmesi için 1977 y›l nda Ron Rivest, Adi Shamir ve Leonard Adleman tarafından geliştirilen RSA algoritmas› kullan›l r. Özet olarak bu algorit-

ma, iki büyük asal sayının çarpımı ile elde edilen sonuçtan ek işlemlerle kamusal ve gizli anahtara karşılık gelen değerlerin üretilmesi esasına dayanır.

1.1. **Kriptolama için Kamusal Anahtar›n Kullan› ›**

Bilginin sahibi tarafından başkalarına bilgi iletileceği zaman, bilgiyi alacak kişinin kamusal anahtar› kullan›l›r. Gönderilecek bilgi al c n n kamusal anahtar ile kriptolanır. Kamusal anahtar alıcı tarafından, gönderen tarafa iletilebileceği gibi, yayınlandığı bir dizin sunucundan da indirilebilir. Ancak pratikte yapılan şey, bilgi iletimi sırasında, rasgele bir oturum anahtarı oluşturularak, simetrik kriptolama teknikleri ile bilginin kriptolanmas›, kamusal anahtar›n ise üretilen anahtar n kriptolanarak korunmas› için kullan›lmas›d r. Simetrik kriptolama teknikleri kamusal/gizli anahtar çiftlerinin kullanıldığı asimetrik kriptolama tekniklerine göre daha hızlı çalışırlar. Bu yüzden kamusal anahtar, pratikte, gönderilecek asıl verinin kriptolanmas yerine, söz konusu oturum için üretilen anahtar›n kriptolanmas için kullanılır. Alıcıya, üretilen anahtarla simetrik kripto teknikleri ile kriptolanmış veri ile kamusal anahtar ile kriptolanmış, oturum için üretilmiş anahtar gönderilir.

1.2. **Çözme için Gizli Anahtar n Kullan› ›**

Gizli anahtar, karşılık geldiği kamusal anahtar ile kriptolanmış verinin çözümü için kullanılır. Veriyi alan kullanıcı, çözebildiği bilginin kendisine gönderildiğinden emin olabilir ancak gelen bilginin kimden geldiğinden emin olamaz. Yine pratikte gerçekleşen şey teoriden biraz farklıdır. Gizli anahtar asıl veriyi kriptolamak için üretilmiş oturum anahtarının çözülmesi için kullanılır. Özel anahtar elde edildikten sonra, bu anahtar ile asıl veriye ulaşmak için çözümleme gerçekleştirilir.

İmza için Gizli Anahtar

Eğer gönderen taraf alıcıya bilginin gerçekten kendisi tarafından gönderildiğini ispatlamak isterse *(Bazen yasal olarak bu gerekebilir),* gizli bir anahtar kullanarak bir mesaj› say sal olarak imzalayabilir *(Say sal imza).* Bu imza, el yaz›s› imzan n aksine her seferinde farklı olarak üretilir. Bir mesaj doğrulama algoritması ile, mesajın içeriği üzerinden bir değer hesaplanır, bu değer gizli anahtar ile kriptolanır. Üretilen bu kriptolanmış değer söz konusu mesaj için bir sayısal imzadır. Say›sal imza mesaj n sonuna eklenerek veya ayr› bir dosya olarak mesaj ile birlikte gönderilir. Kullan lan gizli anahtara karşılık gelen kamusal anahtar da ayrıca bu mesaj ile birlikte, yalnız başına veya bir sertifikanın parçası olarak gönderilmelidir. Ancak say sal bir imza ile korunmuş bir mesaj alınan herkes tarafından işlenip okunabilir. Gönderenin kimliğinin ispatı sağlanmış olsa da güvenlik bu aşamada sağlanmaz.

İmza için Kamusal Anahtar

Sayısal olarak imzalanmış bir mesaj alındığında, alıcı doğru kamusal anahtar kullanarak aşağıdaki adımlarla imzanın doğrulamasını gerçekleştirir:

1. Doğru kamusal anahtar kullanılarak, gönderen tarafın bilgi için hesapladığı değer çözülür.
2. Gönderen tarafın mesaj üzerinden hesapladığı değer, alınan mesaj üzerinden tekrar hesaplan r *(Sertifikaların kullanıldığı durumlarda hesaplama için kulla-n›lan algoritma, sertifika içinde belirtilir).*
3. Yeni hesaplanan değer, birinci aşamada çözülen değer ile karşılaştırılır; eğer değerler uyuşuyorsa, mesaj, söz konusu kamusal anahtara karşılık gelen gizli anahtara sahip taraf tarafından gönderilmiş demektir. Ayrıca iletim sırasında bilginin değiştirilip değiştirilmediği de anlaşılmış olur.
4. Eğer bilgi ile birlikte bir kamusal anahtar sertifikası gönderilmişse, sertifikayı veren güvenlik otoritesinden *(Certificate Authority, CA)* sertifikanın doğruluğu için onay alınır; böylelikle gizli anahtarın sahibinin kimliği de doğrulanmış olur.
5. Son olarak eğer varsa, işlenmiş sertifikanın halen geçerli olup olmadığı kontrol edilir. Eğer vadesi dolmuş ise, ne zaman vadesinin dolduğuna bak›l r.

Bir Word döküman›n n e-mail ile gönderileceği durumu ele alalım. Gönderen taraf söz konusu Word dökümanı için bir hash değeri hesaplar ve gizli anahtarı ile bunu kriptolar. Alıcı, aldığı Word dökümanı için aynı hash değerini hesaplar, gönderen taraftan gelen hash değerini çözüp kendi hesapladığı ile karşılaştırır. Eğer değer aynı ise, gönderenin kimliğinden ve iletim sırasında dökümanın değiştirilmediğinden emin olunması sağlanır. Aksi takdirde, göndericinin kimliği belirsiz ve doküman iletim sırasında değiştirilmiş olacaktır.

Aşağıdaki tabloda kamusal ve gizli anahtarların ne zaman, kim tarafından kullanıldığı gösterilmektedir:

Tablo 1. Kamusal ve gizli anahtar›n kullan› ›

Anahtar İşlevi	Anahtar Türü	Kimin Anahtarının Kullanıldığı
Bir al›c için verinin kriptolanma ›	Kamusal Anahtar	Al› ›
Veri imzalama	Gizli Anahtar	Gönderen
Al nan verinin çözülmesi	Gizli Anahtar	Al› ›
İmzanın doğrulanması	Kamusal Anahtar	Gönderen

Kendi kullan›m›n zdaki bilgileri kriptolamak için, kendi kamusal anahtar›n›z kullanmalısınız; başkasının kamusal anahtarı ile kriptolayacağınız bilgi, sadece o kamusal anahtara karşılık gelen gizli anahtarın sahibi tarafından okunabilir. Bazı sistemler tedbir olarak kriptolama gerçekleştikten sonra as›l döküman tutarlar. Kimi sistemler de karşı tarafın kriptolamak için kullandığı kamusal anahtarını kaydederler.

Genel olarak Kamusal Anahtar Kriptografisi, kriptolama/çözme ve imzalama/doğrulama amacı ile kullanılmaktadır. Kriptolama iletilen bilginin gizliliğini sağlarken, imzalama bilginin kaynağının doğrulmasını sağlar. Ancak unutulmamal -

dır ki, iletilen bilginin sadece imzalanmış/kriptolanmış bölümü korunur. Genellikle e-mail sistemlerinde, e-mail başlıkları bu şekilde korunmazlar.

Hashing

Hashing, bir dizi anlaml› karakterin, matematiksel bir algoritma kullan larak daha kısa ve sabit uzunluklu bir karşılığının elde edilmesidir. Elde edilen kısa karakter dizisi, as›l diziyi temsil edebilmektedir. Bu yöntem özellikle veritabanlar nda indekslerin oluşturulmasında sıklıkla kullanılmaktadır. Örneğin bir veri tabanında ad ve soyadlara karşılık gelen hash dizileri hesaplanıp, sorgulama sırasında girilen ad ve soyad bilgisi hash algoritmas› ile k›salt›ld ktan sonra indeks üzerinde çok daha k sa süren sorgulama işlemleri gerçekleştirilebilir. Uzun ad ve soyad dizileri yerine bunların hash'lenmiş karşılıklarının kullanılması çok daha kısa sürede sorgulamaların sonuçlandırılmasını sağlayacaktır. Hash algoritmalarının, hash değerinden asıl diziyi üretmeleri söz konusu olmadığı gibi bu ayrıca istenen bir durum değildir. İyi bir hashing algoritmas›n›n ayr›ca farkl› iki karakter dizisi için ayn hash dizisini üretmemesi gerekir *(Aksi takdirde bir çarpışma, Collision, oluşur)*. Aşağıda tanımlı basit hash fonksiyonlar› bulunmaktad r:

- **Bölme-kalan yöntemi:** Tablodaki eleman say›s› tahmin edilir. Bu say› her as l değer veya anahtardan bir bölüm ve kalan bulunması için bir bölen olarak kullanılır. Kalan hash'lenmiş değerdir *(Bu yöntem çok sayıda çarpışmaya sebep olabilir).*
- **Katlama:** Bu yöntem asıl değeri *(Bu durumda rakamlar)* birkaç parçaya böler, parçalar› toplar ve son dört rakam *(veya uygun rasgele dört rakam)* hash değeri veya anahtar› olarak kullan r.
- **Taban dönüşümü:** Değer veya anahtar sayısal ise sayı tabanı değiştirilir ve elde edilen yeni değerde belirlenen rakam adedi üzeri rakam göz ardı edilerek hash elde edilir. Örneğin Onluk düzende bir sayı onaltılık düzene çevrilerek bu işlem gerçekleştirilebilir.
- **Say›lar n yeniden düzenlenmesi:** Bu yöntemde asıl değerden rasgele say lar seçilir, sıraları değiştirilerek bir hash değeri elde edilir.

Veri tabanlar›nda kullan›lan tüm hash fonksiyonlar kriptografi veya hata tespiti yöntemleri için uygun olmayabilir. Kriptografi için kullan lan popüler hash fonksiyonlar aras nda MD2, MD4 ve MD5 *(Message Digest, Mesaj Özetleme)* bulunmaktadır. Bu fonksiyonlar sayısal imzaların daha kısa bir değere indirgenmesi için kullanılırlar. MD2, MD4 ve MD5 128 bitlik hash değerleri üretirler. MD2 8 bitlik bilgisayarlar için tasarla ›şken, diğerleri tasarlanırken 32 bitlik bilgisayarlar göz önünde bulundurulmuştur. İçlerinde en güvenilir ancak yavaş olan MD5'dir. SHA *(Secure Hash Algorithm)* MD4'e benzer bir şekilde çalışır ve yine görece geniş bir hash değeri *(60 Bit)* üretir.

Sertifika

Bir sertifika, bir CA tarafından sayısal olarak imzalanmış, bir kamusal anahtar hakkındaki bilgidir. Sertifika içinde bulunan bilgi, ITU X.509 v3 şekline uyar. Bu stan-

darda uyan sertifikalarda, kamusal anahtarın karşılık geldiği gizli anahtarın sahibinin kimliği, anahtar uzunluğu, kullanılan algoritma, kullanılan hashing algoritması, geçerlilik süresi ve sertifikanın hangi amaçlarla kullanılabileceği bilgisi bulunur. Sertifikalar, PKI işletimi için gerekli olmamakla birlikte bazı uyarlamalarda gizli anahtarın sahibinin kimliğinin belirlenmesi gereklidir ve bu amaçla yaygın olarak X.509 şekli kullanılır.

1.3. **Anahtar Kullan›m›n n Denetlenmesi**

Kamusal anahtar sertifikas›ndaki sahalardan biri de anahtar kullan›m sahas›d r. CA taraf›ndan, CA' n hangi tür kullanımları onayladığının belirtilmesi için kullanılır. Bu karşılık gelen gizli anahtarın başka amaçlarla kullanılamayacağı anlamına gelmez; zira gizli anahtarlar için bir sertifika bulunmaz. Kamusal anahtar ile korunmuş bilgiyi alan kullan›c lar, sertifika sağlandığı durumlarda, güncel anahtar kullanımının, sertifikada belirtilen kullanım çeşitlerine uyup uymadığını kontrol etmelidirler.

1.4. **Kamusal ve Gizli Anahtarlar›n Depolanm ›**

Sertifikalar

Kamusal anahtarlar CA'ler taraf›ndan say sal olarak imzalanmış sertifikalar içinde yukar›da bahsedilen bilgiler ile birlikte saklan›rlar. Kamusal anahtarlar herkese aç k olduğundan, okunmalarını engellemenin gereği yoktur; ancak değiştirilmelerini engellemek için say›sal imzalama yöntemi kullan›l r.

Koruma

Gizli anahtarlar, bilgisayarlarda genellikle şifreler ile korunan dosyalarda tutulurlar. Böylelikle ağ üzerinden erişilmiş sistemlerde gizli anahtarların tutulduğu dosyalara erişilse dahi bu şifreler olmadan anahtarlar kullanılamazlar. Anahtarların bilgisayarlarda kullanılması için değişik şekiller kullanımdadır. Örneğin Entrust EPF formatını kullan›rken, Verisign ve GlobalSign P12 format›n› kullanmaktad rlar.

2. **PKI'in Bileşenleri**

Kamusal Anahtar Altyap›s› bir dizi hizmet ve teknolojinin kullan›lmas ile oluşturulmuştur:

2.1. **Sertifika Otoritesi** (CA)

CA'lar sertifikaları düzenler ve doğrulanmasını gerçekleştirirler. CA'lar sertifika talebinde bulunan kurumun veya kişinin kimliğinin ve sertifikanın içeriğinin, sertifika düzenlenmeden önce doğrulanmasından sorumludurlar; bu bilgilerin geçerliliğinin doğru olduklarından emin olunca, sertifikayı düzenler ve sayısal olarak imzalarlar.

Anahtar çiftlerinin üretilmesi

CA kamusal ve gizli anahtar çiftini kendi üretebileceği gibi, sertifika başvurusunu yapan kişi veya kurum kendi anahtar çiftini ürettikten sonra, kamusal anahtar›n içeren imzalanmış sertifika talebini CA'ye doğrulama için gönderebilir. İkinci yöntem gizli anahtar hiçbir zaman kişi veya kurumun denetimi dışına çıkmadığından görece daha güvenli bir yöntemdir.

Sertifikalar n düzenlenmesi

Kurum veya kişi kendi sertifikasını üretmediği sürece *(Baz› yaz›l mlar buna olanak tan›maktad r)*, sertifikalar genellikle CA'den sat›n al›n›rlar. Sertifika CA taraf ndan düzenlenmeden önce, talep gerçekleştiren kurum veya kişinin kimliğinin doğrulanması için bir dizi işlem gerçekleştirilir.

CA, nüfus idareleri veya pasaport dairelerinin PKI karşılığı olarak düşünülebilirler. CA, sertifikan›n düzenlenmesi için gerekli bilgileri ald ktan sonra sertifikadaki bilgilerin değiştirilmesini engellemek için sayısal olarak sertifikay imzalar.

CA'ler ayrıca değişik sınıflarda sertifikalar düzenleyebilirler. Genel olarak üç veya dört tür sertifika bulunmaktad›r. Birinci s›n›f sertifikalar yaln zca geçerli bir e-mail adresi sunularak satın alınabilirler. İkinci sınıf sertifikalarda ise ek olarak bazı kişisel bilgiler istenir. Üçüncü sınıf sertifikalarda ise kurum ve kişinin kimliğinin tespiti için bazı belgeler istenir. Dördüncü sınıf sertifikalar ise yüksek güvenliğin gerektiği hükümet kuruluşları için üst seviyede denetimlerden sonra düzenlenirler.

Sertifikalar›n kullan› ›

Bir kurum veya kişi, farklı CA'lerden alınmış birden fazla sertifikaya sahip olabilir. Baz› Web uygulamalar› kullan›lacak sertifikan›n belirli bir CA taraf ndan üretilmiş olmasını gerektirebilirler. Örneğin bir banka, kendi Web hizmetlerine erişilebilmesi için sadece kendisi tarafından düzenlenmiş sertifikaların kullanılmasını şart koşabilir. Birçok kamusal site ise sunulan her türlü sertifikay kabul edebilir. CA kurum içinde bir birim, bir kurum *(Banka veya PTT)* veya bağımsız bir kuruluş olabilir *(Verisign gibi...)*.

Sertifikaların doğrulanması

CA tarafından sayısal olarak imzalanmış kamusal anahtar sertifikası, sertifika içeriğinin değiştirilmesini engeller. Bu imza ayr›ca sertifikan n halen geçerli olup olmadığının kontrolü için kullanılır. İmza, PKI destekli uygulamalar tarafından, bulunduklar› "Kök CA" listesindeki CA'lar ile denetlenir. Baz› CA sertifikalar tüm sertifika onayları için kaynak teşkil ettiklerinden "Kök Sertifikalar" olarak adland›r› l›rlar. Sertifika onaylama işlemi, kök CA listesinde içerilen uygun kamusal sertifika kullan larak otomatik olarak gerçekleştirilir.

PGP (Pretty Good Privacy) kullan›c›lar› kendi sertifikalar›n düzenlendiklerinden, kendilerini kim olarak tan›t›rlarsa, al›c› taraf bunu kabul etmek durumundad r.

2.2. **Vadelendirme - İptal**

Sertifika kullan›m›nda üzerinde durulmas› gereken konulardan biri de sertifikalar n vadelendirilmesi, geçerlilik sürelerinin belirlenmesidir. Sertifikan n süresi dolduğunda temel olarak iki şey yapılabilir. İlk olarak süresi dolan sertifika yayınlandığı dizinden silinebilir. Bu durumda sertifika sorgulamas› yapanlar, sertifikan n süresinin dolduğu ve iptal edildiği sonucuna varacaklardır.

Ancak bu yaklaşımla ilgili iki sorun bulunmaktadır. Öncelikle sertifikanın yayınlandığı dizin sunucusuna DoS saldırısı düzenlendiğinde *(Sunucu cevap vermez duruma gelene kadar trafik bombardımanına maruz bırakıldığında)*, kullan›c lar cevap alamad›klar nda sertifikanın iptal edilmiş olabileceği sonucuna varabilirler. Ayrıca sertifika sunucunda bilgi silinmesi veya güncellenmesi, sunucunu cevap verme süresini olumsuz bir şekilde etkileyebileceği gibi kullanıcılar sertifika bilgisinin neden dolayı silindiği konusunda bilgi alamayacaklardır. Bu yüzden iptal edilmiş sertifikalar ayrı bir dizinde tutulurlar. Sertifika iptali için CRL *(Certificate Revocaiton List, Sertifika İptal Listesi)* veya OCSP *(Online Certificate Status Protocol, Çevrimiçi Sertifika Durum Protokolü - RFC 2560)* yöntemleri kullan›l r.

2.3. **Kay t Otoritesi**

CA, sertifika taleplerinde kimlik tespiti için Kay t Otoritesi *(Registration Authority, RA)* olarak adlandırılan üçüncü bir tüzel kişiliği kullanabilir. RA, talepte bulunan kuruma CA gibi görünebilir fakat aslen sertifika imzalama işlemini gerçekleştirmez.

2.4. **Sertifika Yay nlama Yöntemleri**

PKI mimarisinin temellerinden biri de kullanıcıların sertifikalara erişiminin sağlanmas› için, sertifikalar›n yay›nlanmas›d r. Bunun için iki yol vard r; sertifikan n bir dizin *(Directory)* sunucunda yay›nlanmas ve e-mail gibi bir yolla yine elektronik olarak ulaştırılması.

Dizin sunucul ›

Dizinler X.500/LDAP uyumlu veritabanlar›d›r. X.509 format ndaki sertifikalara IETF LDAP standard›nda tan›ml arama/sorgulama işlevleri ile erişilebilir. Dizinler kamuya açık olduğu gibi sadece kurum içi erişime de açık olabilirler.

Veritabanl ›

Uygulamaya has veritabanlar› da X.509 format›ndaki sertifikalar›n depolanmas ve kullan›c›lara sunulmas› için kullan labilirler.

E-mail ve diğer elektronik saklama ortamları

Sertifikalar e-mail veya floppy diskler gibi diğer elektronik saklama ortamlarında saklanabilir ve taşınabilirler.

2.5. **Sertifika Yönetim Sistemi**

Bu ifade, sertifikaların yayınlandığı, geçici veya sürekli olarak askıya alındığı, yenilendiği veya iptal edildiği yönetim sistemi için kullanılır. Bu sistemler normalde sertifikalar silmezler.

2.6. **PKI Uyumlu Uygulamalar**

PKI uyumlu uygulamalar belirli bir CA ile PKI işlevlerini yerine getirmek için gerekli yaz l›m eklentilerine sahip uygulamalar› tan›mlamak için kullan›l›r *(Internet Explorer gibi)*.

3. **SSL**

Internet yaşama nüfuz ettikçe, Internet üzerindeki uygulamaların sayısı ve çeşidi de hızla artmaktadır. Bir haberleşme ortamı olmasının yanı sıra, Internet, bir elektronik pazar yeri de olmaktad›r. Internet'in önemi art›kça yap›s ndan kaynaklanan, çözülmesi gereken problemler ortaya çıkmaktadır. Güvenlik bunların başında yer almaktadır; korunması gereken kişisel bilgilerin Internet üzerinden aktarılacağı durumlarda, bankacılık işlemleri ve para transferleri gibi, yeterli güvenliğin sağlanması elzemdir.

WEB sayfalarında, veya html dökümanlarında bağlantılar A *(Anchor)* etiketleri *(Tag)* ile belirtilirler. Tipik olarak bir html dökümanından başka bir dökümana veya siteye link `<a href="http://yasin.kaplan.net/">` şeklinde verilir. Bu tür bir bağlantıya kliklendiğinde tarayıcı *(Internet Explorer-IE veya Netscape Navigator-NN)* söz konusu sunucuya 80 numaralı TCP portu üzerinden bir veya birkaç bağlantı açar. A etiketlerinde 80 numaralı port üzerinde çalışan http protokolü dışında başka protokoller de kullan›labilir. Bu durumda taray›c program söz konusu protokole uygun yaz›l›m parças›n veya müstakil programı çalıştıracaktır. SSL bağlantıları, A etiketinde, https ile başlayan URL'ler kullanılarak belirtilir. SSL, TCP/443 üzerinde çalışır. IE ve NN'ın SSL'e dahili desteği mevcuttur. Bir SSL bağlantısı yapılacağı zaman genellikle taray›c program bir uyarıda bulunur, bağlantı gerçekleştirildiğinde, örneğin IE'de bu *(Sürüm 5.x)* sağ alt köşede kapalı bir asma kilit sembolü ile belirtilir. Asma kilit sembolü üzerine kliklenerek detaylı bilgi alınabilir. SSL bağlant›s› kurulurken, kullan›c ve sunucu arasında bir dizi uzlaşı işlemi gerçekleştirilir; uzlaşı başarısız olursa SSL oturumu kurulmaz, hiçbir bağlantı yapılmaz ve veri iletimi de gerçekleşmez.

4. **SSL Mimarisi**

SSL, TCP üzerinde uçtan uca güvenli bir bağlantı hizmeti sunulması için tasarlanmıştır ve katmanlı bir yapıya sahiptir. SSL Kayıt Protokolü çeşitli üst seviye protokollerine temel güvenlik hizmetleri sunar. WEB sunucusu ve kullan›c› aras nda etkileşimi sağlayan HTTP protokolü SSL üzerinde çalışabilir. SSL parçaları olarak ta-

nımlanmış olan üç adet üst seviye protokol bulunmaktadır: Uzlaşı Protokolü, CipherSpec Değişim Protokolü ve Uyarım Protokolü. SSL Oturumu ve SSL bağlant›s›, iki önemli SSL kavram›d r.

- **Bağlantı:** Sunucu/Kullanıcı arasında uygun türde bir hizmeti sağlayan mantıksal bir bağlant›d r *(Bir web sayfas›n n indirilmesi gibi).* SSL için bunlar uçtan uca ilişkilerdir. Bağlantılar geçicidir. Her bağlantı bir oturuma ilişkilendirilmiştir.
- **Oturum:** Sunucu ve kullanıcı arasında bir ilişkilendirmedir. Oturumlar Uzlaşı Protokolü ile kurulurlar. Oturumlar, birden fazla bağlantı arasında kullanılabilen kriptografik güvenlik parametrelerini tan mlarlar. Oturumlar her bir yeni bağlantı için yeni güvenlik parametre uzlaşı işlemlerinin tekrarlanmasını engellerler.

İki taraf arasında *(Sunucu ve kullan›c üzerinde http gibi uygulamalar...)* birden fazla güvenli bağlantı kurulabilir. Teorik olarak ayrıca birden fazla oturumun da kurulması mümkündür ancak pratikte kullanılan bir özellik değildir.

Her oturumla ilişkilendirilmiş çeşitli durumlar bulunmaktad r. Bir oturum kurulduğunda anlık olarak okuma ve yazma *(Alış-Veriş)* için işletim durumları vardır. Ayr ca Uzlaşı Protokolü etkin iken bekleyen *(Pending)* okuma ve yazma durumlar oluşturulur. Uzlaşı Protokolünün başarılı bir şekilde işletilmesinden sonra bekleyen okuma ve yazma durumları, güncel durumlar haline gelirler. Bir oturum durumu aşağıdaki parametrelerle tan›mlan r:

- **Oturum Belirteci:** Sunucu tarafından seçilen, etkin veya kaldığı yerden devam edilebilen bir oturum durumunu belirtmek için kullan lan rasgele bir Byte dizisi.
- **Eşlik Sertifikası:** SSL oturumundaki uçlardan birine ait bir X.509.v3 sertifikas . Bu durum birimi hükümsüz olabilir.
- **Sıkıştırma Yöntemi:** Kriptolama işleminden önce veriyi sıkıştırmak için kullan lacak algoritma.
- **CipherSpec:** Kullan lacak kriptolama *(DES gibi)* ve karıştırma algoritmasını *(MD5 veya SHA-1 gibi)* tanımlar. Ayrıca karıştırma boyu gibi parametreleri de belirtir.
- **Ana Şifre:** Sunucu ve kullanıcı arasında paylaşılan 48 Byte uzunluğunda bir şifredir.
- **Sürdürülebilirlik:** Oturumun yeni bağlantıları başlatmak için kullanılıp kullanılamayacağını belirtmek için kullanılan bir sahadır.

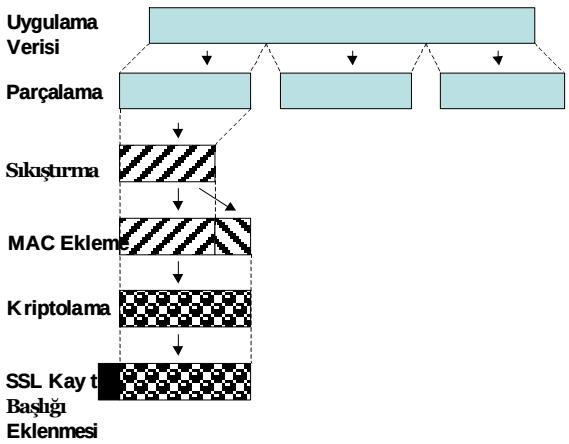
Bir bağlantı durumu aşağıdaki parametrelerle tanımlanır:

- **Sunucu ve kullan›c› rasgele say›s :** Her bir bağlantı için sunucu ve kullanıcı taraf ndan rastgele seçilen Byte dizileri.
- **Sunucu yazma MAC şifresi:** Sunucu taraf ndan gönderilen veri üzerindeki MAC işlemlerinde kullanılan gizli anahtar.
- **Kullanıcı yazma MAC şifresi:** Kullan›c› taraf ndan gönderilen veri üzerindeki MAC işlemlerinde kullanılan gizli anahtar.

- **Sunucu yazma anahtar :** Sunucu taraf›ndan kriptolanan ve kullan›c› tara›ndan çözülen veri için kullan lan konvansiyonel gizli anahtar.
- **Kullan›c› yazma anahtar :** Kullan›c› taraf›ndan kriptolanan ve sunucu tara›ndan çözülen veri için kullan lan konvansiyonel gizli anahtar.
- **Başlatım vektörleri:** CBC *(Chiper Block Chaining)* çalışma şeklinde bir blok şifre kullanılırsa, her anahtar için bir Başlatım Vektörü *(Initialization Vector, IV)* tutulur. Daha sonra nihayi şifre metin bloğu bir sonraki kayıt için IV olarak kullan›lmak üzere saklan r.
- **S›ra numaralar :** Bağlantının her iki ucundaki taraflar, her bağlantıda alınıp verilen mesajlar için ayrı sıra numaraları tutarlar. Bir uç ChiperSpec Değişim mesajı aldığında, uygun sıra numarası sıfıra ayarlanır.

SSL Kay t Protokolü

SSL Kayıt Protokolü SSL bağlantıları için iki hizmet sunar. Uygulama verisinin kriptolanması ile güvenlik, bir mesaj doğrulama kodu *(Message Authentication Code, MAC)* kullanılarak doğruluk sağlanır. Kayıt Protokolü, SSL'in bazı üst protokolleri taraf›ndan kullan labilen temel bir protokoldür. Bunlardan biri, daha sonra anlatılacak olan ve kriptolama ve doğrulama anahtarlarının alış verişi için kullanılan Uzlaşı Protokolüdür.



Şekil 1. SSL Kayıt Protokolü İşletimi

Yukarıdaki şekilde SSL Kayıt Protokolünün nasıl çalıştığı gösterilmektedir. Protokol, iletilecek uygulama mesaj›n› al›p, yönetilebilir parçalara ay rd ktan sonra, seçime bağlı olarak sıkıştırır, bir MAC uygular, bir başlık ekler ve elde edilen paketi bir TCP yığını olarak gönderir. Alınan veri, çözülür, doğrulanır, sıkıştırılmışsa açılır ve tarayıcı gibi uygulama programına ulaştırılır.

İlk adım olan parçalamada her üst katman mesajı, 214 Byte'lık veya daha az uzunlukta bloklara bölünür. Sıkıştırma seçime bağlı olarak bu parçalara uygulan r. SSLv3'de *(veya TLS'de)* herhangi bir sıkıştırma algoritması belirlenmediğinden bu

seçenek hükümsüzdür. Ancak özel uyarlamalar bir sıkıştırma algoritmasını içerebilir.

Bir sonraki adım sıkıştırılan veri üzerinden bir mesaj doğrulama kodunun hesaplanmasıdır. Bu amaçla paylaşılan gizli bir anahtar kullanılır. Aslen karıştırma kodu *(MD5 gibi)*, mesaj, gizli anahtar ve bir miktar dolgu verisi üzerinden hesaplanır. Alıcı aynı hesaplamayı gerçekleştirir ve gelen MAC değeri ile bu sonucu karşılaştırır. Eğer iki değerde birbirine uyuyorsa alıcı iletim sırasında mesajın değişmediğinden emin olur. İletim sırasında bir *hacker* MAC'i üretmek için gerekli gizli anahtar bilmediğinden hem mesajı ve hem de MAC'i değiştiremeyecektir.

Daha sonra sıkıştırılmış mesaj parçası ve ek olarak MAC simetrik kriptolama ile kriptolanır. Bu aşamada DES *(Data Encryption Standart)* ve türevleri kullanılabilmektedir.

Son işlem olarak aşağıdaki sahaları içeren başlık, kriptolanmış bloğa eklenir:

- **İçerik Türü** *(Content Type, 8 Bit)*: İçerikteki mesaj parçasının hangi üst katman protokolü kullanılarak işleneceğini belirtir.
- **Majör Sürüm** *(8 Bit)*: Kullanılan SSL sürümünün majör numarasını belirtir. SSLv3 için bu değer 3'tür.
- **Minör Sürüm** *(8 Bit)*: Kullanılan SSL sürümünün minör numarasını belirtir. SSLv3 için bu değer 0'dır.
- **Sıkıştırılmış Uzunluk** *(16 Bit)*: Mesaj parçasının Byte olarak uzunluğu *(Eğer sıkıştırma kullanılmışsa sıkıştırılmış uzunluk belirtilir)*.

İçerik türü olarak belirlenenler, `change_cipher_spec`, `alert`, `handshake` ve `application_data`'dır. İlk üçü SSL'e has protokollerdir. Uygulama Verisi *(`application_data`)* normalde TCP ile taşınabilecek ancak SSL ile taşınan, uygulamadan gelen veridir. Örneğin SSL ile taşınan HTTP verisi gibi.

CipherSpec Değişimi Protokolü

CipherSpec değişimi SSL'e has ve en basit olan, SSL Kayıt Protokolüne dayanan protokoldür. Bu protokol değeri 1 olan tek Byte'lık bir mesajdan oluşur. Bu mesajın tek amacı bekleyen durumun o anki durum üzerine kopyalanmasını sağlamaktır. Bu işaret bir koordinasyon işareti olarak kullanılır. Kullanıcı taraf ndan sunucuya ve sunucu tarafından kullanıcıya gönderilmelidir. Karşılıklı olarak tarafların bu işareti almasından sonra takip eden tüm mesajlar üzerinde anlaşılan şifreleme ve anahtarlar *(CipherSuite)* ile alınır p verilirler.

Alarm (Alert) **Protokolü**

Alarm Protokolü SSL ile ilgili alarmların uçlara taşınması için kullanılır. Diğer uygulamalarda olduğu gibi, o anki durumda belirtildiği gibi sıkıştırılıp, kriptolanır.

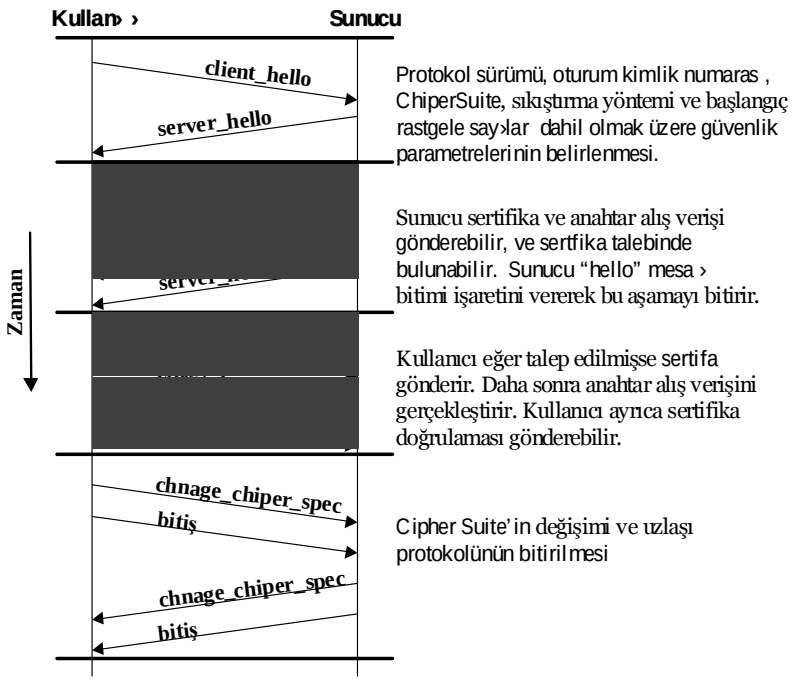
Bu protokolde her mesaj iki Byte'dan oluşur. İlk Byte, Uyar - 1 *(Warning)* veya Ölümcül - 2 *(Fatal)* değerleri ile mesajın önceliğini belirtir. Eğer değer 2 ise, SSL

bağlantıyı hemen keser. Aynı oturumda kurulmuş diğer bağlantılar devam edebilir ancak yenileri kurulmayabilir. İkinci Byte ise detayı belirten bir kod içerir. Örneğin ölümcül mesaj `illegal_parameter` *(Uzlaşı mesajındaki bir saha, dizi dışı veya diğer sahalarla tutarsız bir değer içeriyorsa)* olabilir. Uyarı mesajına örnek ise `close_notify`'dır *(Alıcı tarafa, gönderen tarafın bu bağlantı üzerinden başka mesaj göndermeyeceğini belirtir; her uç bağlantının yazma tarafını kapatmadan önce `close_notify` mesajını göndermesi gerekir)*.

Uzlaşı (Handshake) **Protokolü**

SSL'in en karmaşık bölümü Uzlaşı Protokolüdür. Bu protokol, kullanıcı ve sunucunun birbirlerini doğrulamalarını, SSL kaydı içinde gönderilecek verinin korunması için kullanılacak kriptolama, MAC algoritması ile kriptografik anahtarların belirlenmesini sağlar. Uzlaşı Protokolü, herhangi bir uygulama veri iletilemeden önce kullanılır. Bu protokol sunucu ve kullanıcı arasında alınıp verilen bir dizi mesajdan oluşur.

Aşağıdaki şekilde sunucu ve kullanıcı arasında mantıksal bir bağlantının kurulması için adımlar yer almaktadır. Mesaj alış verişi dört aşamada incelenebilir.



Şekil 2. Uzlaşı Protokolü işletimi. Gri ile işaretlenmiş adımlar seçime bağlıdır.

İlk aşamada mantıksal bir bağlantı başlatılır ve oturum ile ilişkilendirilecek güvenlik parametreleri kurulur. Alış veriş aşağıdaki parametrelerle `client_hello` mesajını gönderen kullanıcı ile başlatılır:

- **Sürüm:** Kullan›c› taraf›ndan alg lanabilen en yüksek SSL sürümü.
- **Rasgele:** Kullan›c› taraf›ndan üretilen ve 32 bitlik zaman damgas ile bir güvenli rasgele sayı üreteci tarafından üretilen 28 Byte'lık bir sayıdan oluşur. Bu değerler Replay saldırılarının engellenmesi için kullan›l r.
- **Oturum kimlik numaras :** Değişken uzunluklu bir oturum belirtecidir. Sıfır olmayan bir değer kullanıcının var olan bir bağlantının parametrelerini değiştirmek istediğini veya söz konusu oturum üzerinde yeni bir bağlantı oluşturmak istediği anlamına gelir. Sıfır olan bir değer kullanıcının yeni bir oturum üzerinde yeni bir bağlantı kurmak istediğini gösterir.
- **CipherSuite:** Azalan öncelikle sıralanmış, kullanıcı tarafından desteklenen kriptolama algoritmalar›n n bir kombinasyonunu içeren bir listedir. Listenin her eleman *(Her CipherSuite)* bir anahtar alış veriş algoritması ve bir CipherSpec'i içerir.
- **Sıkıştırma yöntemi:** Kullanıcının desteklediği sıkıştırma yöntemleri.

`Client_hello` mesaj›n› gönderdikten sonra kullan›c›, ayn parametreleri içeren `server_hello` mesaj›n bekler. `Server_hello` mesaj›nda, Sürüm sahas›, kullan›c taraf›ndan önerilen sürümün minör numaras›n›n sunucu taraf ndan desteklenen en yüksek sürümü içerir. Rasgele sahas› sunucu taraf›ndan üretilir ve kullan›c›n nkinden bağımsızdır. Eğer kullanıcı mesajında Oturum kimlik numarası sıfırdan farklı bir değer içeriyorsa bu aynen sunucu tarafından kullanılır; aksi takdirde sunucunun kimlik sahası yeni bir oturum için yeni bir değeri taşır. CipherSuite sahası ise, kullan›c›n n önerdiği değerlerden, sunucu tarafından seçileni içerir. Aynı şekilde sıkıştırma yöntemi sahas›nda kullan›c›n›n önerdiklerinden sunucu taraf ndan seçileni yer alacakt r.

CipherSuite parametresinin ilk elemanı anahtar alış verişi yöntemidir *(Bu yöntem, konvansiyonel kriptolama ve MAC için kripto anahtarlarının nasıl alınıp verileceğini belirler)*. Aşağıdaki anahtar alış veriş yöntemleri desteklenmektedir:

- **RSA:** Gizli anahtar, al›c›n›n RSA kamusal anahtar › ile kriptolan r. Al c n n anahtar için bir kamusal anahtar sertifikası erişilebilir kılınmalıdır.
- **Sabit Diffie-Hellman:** Bu, sunucu sertifikas›n n, sertifika otoritesi *(Certificate Authority, CA)* taraf ndan imzalanan Diffie-Hellman kamusal parametrelerini içerdiği, bir Diffie-Hellman anahtar alış verişidir. Kamusal anahtar sertifika ›nda da Diffie-Hellman kamusal anahtar parametreleri bulunmaktad›r. Kullan c n n Diffie-Hellman parametreleri ya bir sertifika ile ya da bir anahtar alış verişi mesajı ile sağlar. Bu yöntemin sonucunda sabit kamusal anahtarların kullan›m ile Diffie-Hellman hesaplamas›na dayanarak iki uç aras nda gizli anahtarlar edinilmiş olur.
- **Geçici Diffie-Hellman:** Bu yöntem geçici gizli anahtarların oluşturulması için kullan›l r. Bu durumda Diffie-Hellman kamusal anahtarlar› al›n p verilir ve gönderen tarafın RSA veya DSS anahtarları ile imzalanır. Alıcı imzayı doğrulamak için karşılık gelen kamusal anahtarı kullanabilir. Sertifikalar kamusal anahtarları doğrulamak için kullanılır. Bu yöntem, üç ayrı Diffie-Hellman seçe-

neği arasında, geçici, doğrulanmış anahtarların üretilmesinden dolayı en güvenli olan›d r.

- **Anonim Diffie-Hellman:** Bu seçenekte temel Diffie-Hellman algoritmas doğrulama olmaks›z›n kullan›l r. Yani uçlar kamusal Diffie-Hellman parametrelerini birbirlerine doğrulama olmaksızın göndermektedirler. Bu seçenek güvenlik aç›s›ndan en zay›f olan›d r.

Anahtar alış veriş yöntemini takiben, CipherSpec ile kriptolama, karıştırma ve ilgili diğer parametreler belirtilir.

Sunucu ikinci aşamayı kendi sertifikasını göndererek başlatır; eğer doğrulanmas gerekiyorsa, mesaj› bir veya bir dizi X.509 sertifikas›n› içerecektir. Sertifika mesaj , Anonim Diffi-Hellman dışında üzerinde anlaşılan tüm anahtar alış verişi yöntemleri için gereklidir. Eğer sabit Diffie-Hellman yöntemi kullan›l yorsa, bu sertifika mesaj sunucunun anahtar alış veriş mesajı olarak işlev göreceğine dikkat edilmelidir zira sunucunun kamusal Diffie-Hellman parametrelerini içerecektir.

Daha sonra eğer gerekiyorsa bir `server_key_exchange` mesajı gönderilebilir. İki durumda gerekli değildir: Sunucu sabit Diffie-Hellman parametreleri ile bir sertifika gönderdiğinde ya da RSA anahtar alış verişi kullanılmadığında.

Bir sonraki ad mda anonim olmayan *(Anonim Diffie-Hellman kullanmayan bir sunucu)* bir sunucu kullan›c dan bir sertifika talep edebilir. `Certificate_request` mesaj iki parametreyi içerir: `certificate_type` ve `certificate_authorities`. Sertifika türü kamusal anahtar algoritmasının türünü belirtir. İkinci parametre kabul edilebilir sertifika otoritelerinin bir listesidir.

İkinci aşamanın son ve her zaman gerekli mesaj `server_done` mesaj›d r. Bu mesaj sunucu taraf ndan `server_hello` ve ilgili mesajların bittiğini belirtmek için kullan l›r. Bu mesaj gönderildikten sonra sunucu, kullan›c› cevab için bekler. Bu mesaj herhangi bir parametre içermez. Kullan›c› taraf ndan `server_done` mesaj› al›n›k tan sonra, eğer gerekli ise kullanıcı sunucunun geçerli bir sertifika sunduğunu doğrulamal ve sunucunun `server_hello` parametrelerinin kabul edilebilir olduğunu kontrol etmelidir. Eğer tüm bunlar tatmin edici ise kullanıcı üçüncü aşamada sunucuya bir veya daha fazla mesajı geri döndürür. Sunucu bir sertifika talep etmişse, kullanıcı yeni aşamaya bir sertifika mesajı göndererek başlar. Eğer uygun bir sertifika yoksa kullan›c bunun yerine `no_certificate` alarm mesaj›n gönderir.

Daha sonra bu aşama içerisinde `client_key_exchange` mesaj gönderilir. Bu mesajın içeriği kullanılan anahtar alış veriş yöntemine göre farklılık gösterir. Üçüncü aşamada son olarak kullanıcı bir kullanıcı sertifikasının açık doğrulamas için `certficate_verirfy` mesaj›n› gönderebilir. Bu mesaj yaln zca imzalama yeteneğine sahip bir kullanıcı sertifikasını takiben gönderilebilir *(Yani sabit Diffie-Hellman d›şında)*.

Dördüncü aşama güvenli bir bağlantının kurulmasıyla tamamlanır. Kullan›c bir `change_cipher_spec` mesaj gönderir ve bekleyen CipherSpec güncel CipherSpec üzerine kopyalanır. Bu mesaj Uzlaşı Protokolünün bir parçası değildir ve

CipherSpec Değişimi Protokolü ile gönderilir. Kullanıcı bitiş mesajını, ivedilikle, yeni algoritmalar, anahtarla ve şifreler altında gönderir. Bitiş mesajı anahtar alış verişi ve doğrulama işlemlerinin başarılı olduğunu bildirir.

Bu iki mesaja cevaben sunucu kendi `change_cipher_spec` mesaj›n gönderir ve bekleyen CipherSpec'i güncel olana kopyalar. Bu aşamada Uzlaşı tamamlanmıştır; sunucu ve kullanıcı uygulama katmanı verisinin alış verişine başlayabilirler.

Uygulama verisi iletildikten sonra TCP oturumu kapat›l r. Ancak TCP ve SSL arasında doğrudan bir bağlantı olmadığından SSL durumu sürdürülebilir. Sunucu ve kullanıcı arasında yapılacak takip eden iletimler üzerinden uzlaşılmış parametreler ile gerçekleştirilebilir. Http için bu durum kullanıcının aynı sunucundan bir bağlantıya klikleyerek başka bir dökümanı talep etmesiyle yaşanır. Böyle bir durumda sunucu ve kullanıcı güvenlik parametreleri için başka bir uzlaşı gerçekleştirmeyecek ve daha önceki parametreler kullanılarak iletim gerçekleştirilecektir. SSL tanımı bu bilgilerin 24 saatten daha kısa bir süre tutulmasını önerir. Eğer bu zaman zarfında yeni bağlantı yapılmaz ise tutulmakta olan bilgiler silinir.

5.**Taşıma Katmanı Güvenliği** (Transport Layer Security, TLS)

TLS, IETF tarafından, SSL'e dayanan, taşıma katmanında genel güvenliği sağlamak için geliştirilmekte olan bir standarttır. Taslak sürümü SSLv3'e çok benzemektedir. Ancak MAC, gizli anahtarların üretilmesi gibi işlemlerde farklı algoritmalar kullanılmaktadır. TLS ile TCP veya UDP'yi kullanan tüm uygulamalar için genel bir işletim modeli sağlanacaktır. TLS'de ayrıca daha fazla sayıda tanımlı uyar› mesaj bulunmaktad r.

6. **Özet**

PKI, her geçen gün güvenlik problemleri artan Internet üzerinde bilgi iletiminde güvenliğin sağlanması için tasarlanmış olan bir güvenlik altyapısıdır. PKI aslen Internet gibi doğasında güvenlik sistemleri tanımlı olmayan ağlar için tasarlanmış olsa da, dahili kurumsal ağlar gibi özel ağlarda da kullanılabilir. PKI'dan ayrıca güvenlik gerektiren farkl› uygulamalar için (VPN uygulamalar gibi) kriptografik anahtarların dağıtılması, ulaştırılması için de faydalanılabilir.

Günümüzde WEB veya WWW (World Wide Web), Internet üzerindeki en bilinen hizmet türüdür ve Internet üzerindeki uygulamalardan tamam›na yak›n bu ortam üzerinde çalışmaktadır. WEB, TCP tabanlı bir hizmettir. Bu tür uygulamalarda güvenliği ağlamanın en yayg n yolu SSL (Secure Sockets Layer, Güvenli Soket Katman›)'dir. IETF ayr›ca SSL ile geriye dönük olarak uyumlu, genel bir standart t »mlamak için oluşturduğu TLS (Transport Layer Securiy, Taşıma Katmanı Güvenliği) isimli bir çalışma grubu da bulunmaktadır. SSL, Netscape tarafından öncülüğü gerçekleştirilmiş ve SSL v3.0 bir Internet taslak dökümanı olarak sunulmuştur. TLS, SSL v3.1 olarak düşünülebilir. SSL, her ne kadar TCP ve uygulama katmanı arasın-

da saydam bir güvenlik sistemi olarak WEB dışındaki protokollere de uygulanabilir olsa da, pratik uygulamaları WEB ile sınırlı kalmıştır.

7. **Sorular**

1. PKI'ın temel çalışma prensibi neye dayanmaktadır?
2. PKI'da say›sal imzalama ne amaçla kullan›lmaktad r?
3. Hashing işlevsel olarak ne sağlamaktadır? Uygulanırken nelere dikkat edilmelidir?
4. PKI'da sertifikalar n rolü nedir?
5. PKI'ın temel bileşenleri nelerdir?
6. PKI'da CA' n rolü nedir?
7. SSL'de bağlantı ve oturum kavramlarını açıklayınız.
8. SSL kayıt protokolü aşamalarını açıklayınız.
9. SSL'de sertifikalar nerede ve ne amaçla kullan›lmaktad rlar?
10. SSL ve TLS aras ndaki farklar nelerdir?