Bulut Bilişim Güvenliği için Homomorfik Şifreleme

Esra Çalık[1], Hüseyin Aşkın Erdem[2], M. Ali Aydın[3]

[1] Fatih Sultan Mehmet Vakıf Üniversitesi, İstanbul

[2] Hava Harp Okulu, Havacılık ve Uzay Teknolojileri Enstitüsü, Bilgisayar Mühendisliği A.B.D., İstanbul

[3] İstanbul Üniversitesi, Bilgisayar Mühendisliği Bölümü, İstanbul

ecalik@fsm.edu.tr, herdem@hho.edu.tr, aydinali@istanbul.edu.tr

Özet: Bu çalışmada temel olarak, bulut bilişim alanında kullanılan verilerin gizliliğini ve güvenliğini sağlamak amacıyla kullanılabilecek yöntemlerden, Homomorfik şifreleme yöntemi tanıtılmıştır. Homomorfik şifrelemenin ne olduğu, çeşitleri ve kullanım alanları bu çalışma kapsamında araştırılmıştır. Çalışmada, şifreleme ve şifre çözme işlemleri, homomorfik olarak Rivest-Shamir-Adleman algoritması üzerinden bir örnekle detaylandırılmıştır. Ayrıca, homomorfik şifreleme yönteminin, bulut bilişimde veri gizliliğini ve güvenliğini nasıl sağlayacağı örnekler ile desteklenerek açıklanmıştır.

Anahtar Sözcükler: Homomorfik Şifreleme, Rivest-Shamir-Adleman Algoritması, Bulut Bilişim, Veri Gizliliği.

Abstract: This study basically aims to introduce homomorphic encryption method which can be used in cloud computing area in order to establish data privacy and security. What the homomorphic encryption means, types and usage areas of homomorphic encryption are explained. Additionally, encryption and decryption processes are explained by homomorphical usage of Rivest-Shamir-Adleman algorithm. Also in this work, how to accomplish data privacy and security in cloud computing by homomorphic encryption is examined.

Keywords: Homomorphic Encryption, Rivest-Shamir-Adleman Algorithm, Cloud Computing, Data Privacy.

1. Giriş

Artan veri depolama ihtiyaçları nedeniyle son zamanlarda sıklıkla karşılaşılan "bulut" kavramı depolanan her tür veriye her an her yerden ulaşma imkânı sağlamaktadır. Bulut kavramı her ne kadar son kullanıcıya çok geniş depolama alanları ve hesaplama kaynakları sunuyor olsa da, bulut bilişim güvenlik ve gizlilik açısından halen sağlam bir çatı altına alınamamıştır.

Bulut bilişimde verilerin, geniş bir ağ şeklinde konumlamış ve üzerinde kullanıcıların kontrolü olmayan dağıtık makineler ve kaynaklardan oluşan bir buluta yüklenmesi söz konusudur. Bu aşamada kullanıcı, bulut sağlayıcısının, verilerini kullanarak yaptığı işlemlerin güvenilirliği konusunda, kaynak sağlayıcıya güvenmek durumunda kalacaktır. Çünkü böyle bir senaryoda, kullanıcının verileri şifresiz olarak işlenmekte ve tüm işlemler sadece kaynak sahibinin kontrolü altında gerçekleştirilmektedir. [1]

İşlemlerin, şifreli olarak gerçekleştirilmesi, oluşan güvensizlik ortamını engelleyebilecek ve veri gizliliğini sağlayabilecektir. Bu noktada, verilerin işlenmesi sırasında homomorfik şifreleme kullanılarak, şifre çözümüne gerek kalmadan işlemler yapılabilmekte ve işlemlerin şifresiz sonucunu sadece kullanıcı görebilmektedir.

Yani, şifreyi çözecek anahtar sadece kullanıcının elinde bulunmaktadır.

İkinci bölümde, homomorfik şifreleme kavramı tanıtılmış ve Rivest, Shamir ve Adleman (RSA) algoritması yardımıyla örneklendirilmiştir. Üçüncü bölümde, bulut bilişim kavramı ve bulut bilişimde homomorfik şifrelemenin nasıl yapılabileceği incelenmiştir. Dördüncü bölümde, RSA algoritması kullanılarak geliştirilen örnek bir homomorfik çarpma işlemi uygulaması açıklanmıştır. Beşinci bölümde, bulut bilişim güvenliği için homomorfik şifreleme çalışması özetlenmiştir.

2. Homomorfik Şifreleme

Homomorfik şifreleme, şifreli metin üzerinde yapılan belirli matematiksel işlemlerin sonucunda oluşturulan şifreli sonucun, şifresi çözüldüğünde, elde edilecek sonuç ile bu işlemlerin açık metin üzerinde uygulanmasıyla elde edilecek sonucun, aynı olmasını sağlayan bir şifreleme türüdür [2].

Homomorfik şifrelemeyi formül ile ifade etmek gerekirse; Enc şifreleme işlemini, Dec şifre çözme işlemini ve K da şifreleme için kullanılan gizli anahtarı, bunların yanında + ve * işaretleri de Q kümesi üzerindeki toplama ve çarpma işlemlerini temsil etsin. Bu durumda, [3]

Enc şifreleme fonksiyonunun; homomorfik toplama özelliği taşıması için
$\forall$ a, b ϵ Q ifadesi ile
$a + b = Dec_K (Enc_K(a) + Enc_K(b))$ eşitliğinin sağlanması, [3]

ve homomorfik çarpma özelliği taşıması için de $\forall$ a, b ϵ Q ifadesi ile
$a * b = Dec_K (Enc_K(a) * Enc_K(b))$ eşitliğinin sağlanması gerekmektedir. [3]

Bu eşitliklerde kullanılan $Enc_K(a)$, a sayısının K gizli anahtarı ile şifrelenmesini, Dec_K ise elde edilen şifreli toplam veya çarpım işlem sonucunun, yine K gizli anahtarı kullanılarak, şifresinin çözülmesini ifade eder.

Bu şekilde, veri içeriğinin gizliliği ve işlemlerin güvenli şekilde tamamlanabilmesi sağlanabilmektedir.

Farklı şirketlerdeki farklı servislerin, homomorfik şifreleme ile işlemlerini gerçekleştirmeleri sayesinde güvenilir bir hesaplama zinciri oluşturulabilmektedir. Örneğin, birinci servis vergilerin hesaplanmasıyla ilgili sorguları, ikincisi döviz değişim oranlarıyla ve üçüncüsü de nakliye işlemleriyle ilgili sorguları işlettiğinde, her servis sadece kendine ait sorguları işletmiş ve diğer servislerin verilerini elde etmemiş olacaktır. [2]

Çeşitli şifreleme sistemlerinde homomorfik şifreleme ile veri gizliliğinin sağlanması; güvenli elektronik oylama sistemlerinin geliştirilmesine (her oy şifrelenir, sadece toplam sonucu deşifre edilir), gizli bilgi erişim yöntemlerine, tıbbi kayıtların tutulmasına, çakışma-dirençli (collision-resisant) hash fonksiyonlarının üretilmesine, veri madenciliğine ve bulut bilişim alanında güvenli işlemlerin gerçekleştirilmesine yardımcı olmaktadır. [2][5][6]

Şifreli mesajlar üzerinde basit işlemler gerçekleştirilmesi fikri ilk olarak Rivest, Adleman ve Derouzous tarafından öne sürülmüştür. Bu şekilde bir yaklaşımın temelinde ise şifreli olarak saklanmış veritabanına erişimin, sadece yetkilendirilmiş kişiler tarafından şifre çözülmeden yapılmasına olanak tanınması bulunmaktadır. [5]

Homomorfik şifreleme, uygulanacak işlem operatörü sayısına göre, kısmi ve tam homomorfik şifreleme olarak ikiye ayrılmaktadır. Kısmi-homomorfik şifrelemede (Partially-homomorphic encryption), şifreli metinler üzerinde gerçekleştirilebilecek tek bir işlem (ya toplama ya da çarpma) söz konusuyken, ilk olarak 2009'da Craig Gentry'nin tez çalışmasında [8] gündeme getirilen tam-homomorfik şifrelemede (Fully-homomorphic encryption) ise hem toplama hem de çarpma işlemlerinin bir arada

uygulanması söz konusudur. Örneğin, RSA ve El-Gamal algoritmaları çarpmaya göre kısmi şifreleme özelliği göstermektedir. [2]

Gahi ve arkadaşları çalışmalarında, tam homomorfik şifreleme kullanarak, güvenli bir veritabanı sistemi geliştirmişlerdir. Önerilen bu model, şifreli verileri giriş olarak almakta ve sonrasında kullanıcı sorgusunun içeriğini bilmeden, "görmeden işleme-blind processing" şeklinde işlemektedir. Şifreli olarak üretilen sonuç da sadece sorguyu başlatan kullanıcı tarafından deşifre edilmektedir. Bu şekilde, sunucuların bütünlükleri (integrity) kuşku duyulabilir olsa bile kullanıcıların gizlilikleri korunmuş ve uzak uygulamalara olan güvenleri sağlanmış olmaktadır. [2][7]

Günümüzde kullanılan homomorfik şifreleme sistemleri; toplama, çıkarma, çarpma, XOR ve üs alma gibi işlemleri gerçekleştirebilmektedir. Bu yöntemler, şirketlerin tüm veritabanlarını şifreleyerek buluta yükleyebilmelerinin ve bu veriler üzerinde şifre çözme işlemleri uygulamadan hesaplamalar yapabilmelerinin önünü açmıştır. [5]

En yaygın kullanılan şifreleme sistemlerinden açık-anahtar (public key) kullanan RSA, bilinen ilk homomorfik şifreleme yöntemlerinden birisidir. Çarpmaya göre kısmi-homomorfik özellik gösteren RSA algortimasında, m şifrelenecek mesaj, n modül, e açık anahtar, d gizli anahtar, E şifreli metin olarak alındığında, yapılacak şifreleme (Enc) ve şifre çözme (Dec) işlemleri aşağıdaki formüllerden elde edilmektedir: [5]

Şifreleme: $Enc(m) = m^e \pmod n = E$
Şifre çözme: $Dec(E) = E^d \pmod n$

Burada kısmi-homomorfik şifreleme işlemi, RSA şifreli metinlerinin ayrı ayrı çarpılmasıyla bulunacak sonucun, şifresiz metinlerin çarpımının şifrelenmiş haline eşitliğini gerektirmektedir:

$Enc(m_1) * Enc(m_2) = m_1^e * m_2^e \pmod n = (m_1 * m_2)^e \pmod n = Enc(m_1 * m_2)$

Örneğin, $m_1 = 4$, $m_2 = 3$, $e = 7$, $d = 3$, $n = 33$ alındığında,

c_1, m_1 verisinin şifreli metni olmak üzere;
$c_1 = Enc(m_1) = m_1^e \pmod n = 4^7 \pmod{33} = 16$

c_2, m_2 verisinin şifreli metni olmak üzere;
$c_2 = Enc(m_2) = m_2^e \pmod n = 3^7 \pmod{33} = 9$

c_1 ve c_2 şifreli metinlerinin çarpımı;
$c_1 * c_2 = 16 * 9 = 144$

($c_1 * c_2$) şifreli metinlerinin çarpımının, şifresi çözülürse;
$Dec(c_1*c_2) = (c_1*c_2)^d \pmod n$
$= (144)^3 \pmod{33} = 12$

m_1 ve m_2 şifresiz metinlerinin çarpımı;
$m_1 * m_2 = 4 * 3 = 12$

$Enc(m_1)*Enc(m_2) = Enc(m_1*m_2)$
sonuçlarının eşitliği RSA kısmi-homomorfik şifreleme için sağlanmış olacaktır. [5]

Bir sonraki bölümde, homomorfik şifreleme ve homomorfik şifrelemenin bulut bilişim güvenliği açısından değerlendirilmesi ele alınmıştır.

3. Bulut Bilişim Güvenliğinde Homomorfik Şifreleme

Bulut bilişim, çevrim içi bilgi dağıtımı amacıyla kullanılan ve çeşitli hesaplama kaynaklarından (sunucular, depolama alanları, uygulamalar, servisler…) oluşan internet-tabanlı bir teknolojidir [4][5]. Bulut bilişim, gerek düşük maliyetli kullanım olanakları sağlaması, gerekse büyük hesaplama olanakları sunması açısından sıklıkla tercih edilmektedir. Ancak, güvenlik ve veri gizliliği açısından bakıldığında burada iki önemli soru karşımıza çıkmaktadır. Bunlardan birincisi "Bulut bilişim veri-merkezleri saldırıya uğradığında, son kullanıcı verisinin elde edilemeyeceğinden nasıl emin olunabilir?" ve ikincisi de "Son kullanıcı verisi nasıl bulut sağlayıcısından

gizli ve görünmez tutulabilir?". Burada ilk akla gelen çözümlerden birisi verilerin bulut üzerine şifrelenerek depolanması olabilir. Ancak, bu veriler üzerinde işlem yapılması gerektiğinde, bulut sağlayıcı tarafından bu şifrelerin çözülmesi, işlemin şifresiz veriler üzerinde gerçekleştirilmesi ve sonrasında sonucun tekrar şifrelenerek kullanıcıya gönderilmesi veya bulutta depolanması gerekmektedir. Böyle bir yaklaşım, hem şifreleme hem de şifre çözme işlemleri için ek hesaplama ve bellek kullanımlarını gerektirecektir. Daha da önemlisi kullanıcı, verilerini şifrelemek ve şifresini çözebilmek için kullandığı gizli anahtarını (private key) bulut sağlayıcısı ile paylaşmak durumunda kalacaktır ki bu da bulutta depolanmış verilerin gizliliğini etkileyecektir.[6]

İşlemlerin homomorfik şifreleme ile gerçekleştirilmesiyle, verilerin şifresi çözülmeden işlemler yapılmakta ve böylelikle hem çözüme ulaşma hızı arttırılabilmekte hem de veri gizliliği sağlanabilmektedir.[6]

Genel olarak, istemci şirket ve bulut sağlayıcı (sunucu) arasında, sunucunun orijinal veriyi hiçbir şekilde görmeden işlemleri gerçekleştirdiği bir veri hesaplama işlemi aşağıdaki gibi özetlenebilir: [6]

İstemci: a ve b verilerini, *Enc*(a) ve *Enc*(b) olarak şifreler ve sonrasında *Enc*(a) ve *Enc*(b)'yi sunucuya gönderir. [6]

Sunucu: *Enc*(a) ve *Enc*(b) şifreli metinlerine belirli bir işlemi (örneğin, matematiksel toplama/çarpma) ifade eden f-fonksiyonunu uygular ve hesapladığı şifreli sonucu f(*Enc*(a), *Enc*(b)) istemciye gönderir. [6]

İstemci: İstenilen hesaplama sonucunu gizli anahtarı ile deşifre eder ve böylelikle, verilerin/işlem sonucunun gizliliği ve güvenliği sağlanmış olur. [6]

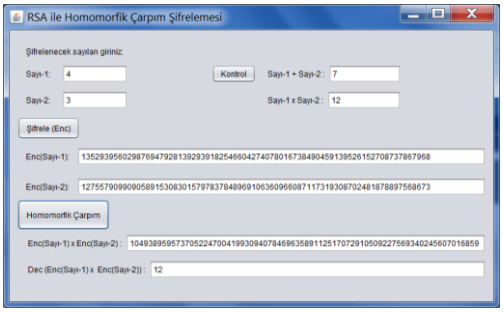
Homomorfik şifreleme ile bir şirket, tüm e-posta veritabanını şifreleyerek bulut üzerine yükleyebilir ve bulutta-depolanmış bu verileri istediği şekilde (örneğin, çalışanların grup/ortak çalışma şekillerini belirlemek için veritabanı araması) sorgulayıp, elde edeceği sonuçları buluttan indirdikten sonra şifrelerini çözerek (e-postaların içeriklerinin teker teker incelenmesine gerek kalmadan) inceleme yapabilir. [5]

Bir sonraki bölümde, RSA algoritmasının kullanıldığı bir homomorfik çarpma işlemi uygulaması anlatılmıştır.

4. Uygulama

Bu çalışmada, RSA Algoritmasının homomorfik çarpma özelliği kullanılarak Java programlama dili ile temel bir homomorfik şifreleme uygulaması oluşturulmuştur.

Hazırlanan uygulamanın arayüzü Şekil.1'de gösterilmiş ve uygulama, Bölüm.2'de verilen sayısal örnekteki değerler kullanılarak çalıştırılmıştır.



Şekil 1. RSA homomorfik şifreleme uygulaması arayüzü.

Uygulama başlatıldığında açılan arayüzde, öncelikle kullanıcıdan iki tane tamsayı girmesi beklenmektedir. Kullanıcının girdiği tamsayılar, `Şifrele (Enc)` butonu ile RSA algoritmasına göre ayrı ayrı şifrelenmektedir. Şifreli sayılar `Homomorfik_Çarpım` butonu ile RSA algoritmasının "homomorfik çarpım" özelliğine göre çarpılmaktadır. Elde edilen ve `Enc(Sayı1)*Enc(Sayı2)` kısmında gösterilen şifreli çarpım verisinin, RSA algoritmasına göre şifresi çözüldüğünde elde edilen sonuç, arayüzün en altında `Dec(Enc(Sayı1)*Enc(Sayı2))` kısmında gösterilmektedir. Gerçekleştirilen şifreli

hesaplamaların sağlamasının yapılması amacıyla, şifresiz sayılar üzerinde yapılan kontrol hesaplamaları da arayüzün sağ kısmında `Kontrol` butonu ile hesaplatılmaktadır.

Bir sonraki bölümde, yapılan çalışma özetlenerek, bulut bilişim güvenliğinin homomorfik şifreleme kullanılarak nasıl sağlanabileceği genel olarak yorumlanmıştır.

5. Sonuçlar

Bu çalışmada, verilerin şifrelerinin çözülmesine gerek kalmadan, şifreli veriler üzerinde işlem yapılmasına olanak sağlayan homomorfik şifreleme yöntemi araştırılmıştır. Homomorfik şifreleme kullanılarak, veri deşifre edilmeden toplama veya çarpma gibi işlemler gerçekleştirilebilmektedir. Bu şekilde, veri içeriğinin saklı kalması (sunucu / uzak kaynak (remote resource) tarafından görülmeden) ile veri gizliliği sağlanmaktadır. Bulut bilişimdeki en büyük açıklardan birisi olan veri gizliliği ve güvenliğinin sağlanamaması, bulut sağlayıcısının, işlemleri, homomorfik şifreleme ile şifreli verilerin üzerinden gerçekleştirmesi ile aşılabilmektedir. Böylelikle kullanıcı, hesaplamaların şifreli sonucunu sadece kendinde bulunan gizli anahtar yardımıyla deşifre edebilmekte ve verilerin güvenliği ve gizliliği sağlanabilmektedir. Bu çalışmada, RSA algoritmasının homomorfik şifreleme özelliği kullanılarak bir uygulama geliştirilmiş ve bu uygulama kullanılarak şifreli veriler üzerinden yapılacak çarpma işlemi sonucunun, şifresiz verilerle gerçeklenecek çarpma işlem sonucuna eşit olduğu gösterilmiştir.

Kaynaklar

[1] Brenner M., Wiebelitz J., Voigt G.V. ve Smith M., "Secret Program Execution in the Cloud Applying Homomorphic Encryption", 5th IEEE International Conference on Digital Ecosystems and Technologies (IEEE DEST 2011), Daejeon, Korea, 2011, pp 114-119.

[2] http://en.wikipedia.org/wiki/Homomorphic_encryption (Son Erişim Tarihi: 14.10.2014)

[3] Özdemir S., "Kablosuz Algılayıcı Ağlarında Homomorfik Şifreleme İle Güvenli Veri Kümeleme", Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, Cilt 23, No 2, 2008, pp 365-373.

[4] Mell P. ve Grance T., The NIST Definition of Cloud Computing-Recommendations of the National Institute of Standards and Technology, NIST Special Publication 800-145, 2011.

[5] Ravindran S. ve Kalpana P., "Data Storage Security Using Partially Homomorphic Encryption in a Cloud", International Journal of Advanced Research in Computer Science and Software Engineering, 3(4), 2013, pp 603-606,.

[6] Tebaa M., Hajji S.E. ve Ghazi A.E., "Homomorphic Encryption Applied to the Cloud Computing Security", Proceedings of the World Congress on Engineering 2012 Vol I WCE 2012, London, U.K., 2012, pp 536-539.

[7] Gahi Y., Guennoun M. ve ElKhatib K., "A Secure Database System using Homomorphic Encryption Schemes". The Third International Conference on Advances in Databases, Knowledge, and Data Applications DBKDA, 2011, pp 54–58.

[8] Craig Gentry, "A Fully Homomorphic Encryption Scheme", Phd. Thesis, Standford University, 2009.