

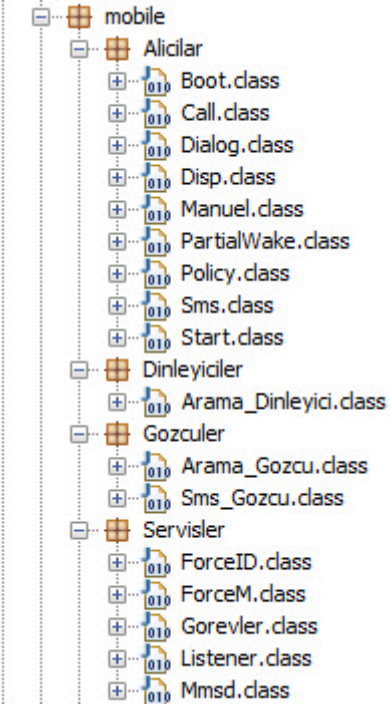
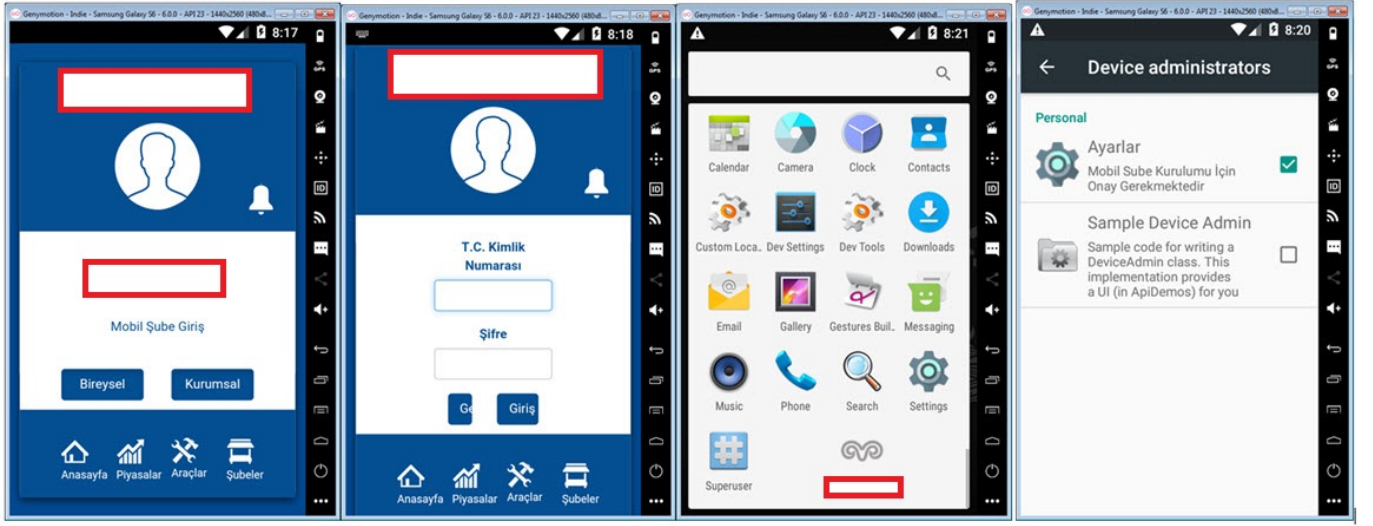
Android Hata Ayıklaması

written by Mert SARICA | 1 April 2018

Yıllar içinde Android işletim sistemi yüklü mobil cihazların pazar payının bir hayli yükselmesi, teknoloji meraklıları kadar zararlı yazılım geliştiricilerinin de dikkatini çekmeyi başardı. Öyle ki günümüzde Windows işletim sistemi kullanıcıları nasıl zararlı yazılımlara karşı ilave güvenlik yazılımları kullanma zorunluluğu hissedip, günlük işleri arasında aldıkları sıradan bir e-postada yer alan bir bağlantı adresine (link) tıklamadan önce 40 kere düşünmek durumunda kalıyorlarsa, Android işletim sistemi kullanıcıları da benzer bir paranoya içinde günlerini geçirmeye başladılar. Nitekim haksız olduklarını söylemem oldukça güç özellikle bankacılık zararlı yazılımlarının gelişimini hem işi hem de özel ilgisi nedeniyle yakından takip eden bir güvenlik araştırmacısı olarak son aylarda karşılaştığım zararlı yazılımlar beni bir hayli şaşırtıyor.

Şöyle dönüp 2012 yılında yayımlamış olduğum Android Zararlı Yazılım Analizi başlıklı blog yazıma baktığımda, o zamanlar Android işletim sistemi için geliştirilmiş olan zararlı yazılımları analiz etmenin çok da zor olmadığını, günümüze kıyasla statik analizin zararlı yazılımları analiz etmede tek başına yeterli olabildiğini görüyorum. Günümüzde ise işin rengi git gide değişip işler, zararlı yazılım analistleri için her geçen gün daha da karmaşık bir hal alıyor.

İlk olarak ~2 yıl önce görülen ve hem Bakır EMRE'nin yazısına hem de benim Android Anti Anti-Emulator başlıklı blog yazıma konu olmuş olan, çağrı dinleme ve SMS mesajlarını okuma özelliklerine de sahip zararlı yazılımın güncel sürümü ile geçtiğimiz aylarda karşılaştığımda, kullanıcı arabiriminin (UI) yıllar içinde geldiği nokta beni fazlasıyla şaşırttı.



Kullanıcı arabirimi bir kenara, uygulamanın kaynak kodunda okunaklı olmayan karakter dizilerinin (strings) çalışma esnasında özel bir fonksiyon ile XOR işleminden geçirilerek çözülüyor olması ve ayrıca bazı önemli değerlerin (ip adresi, tor adresi vs.) AES şifreleme algoritması ile şifreli bir şekilde saklanıp yine benzer yöntemde farklı bir fonksiyon ile çalışma esnasında (run-time) çözülüyor olması, fonksiyon isimlerinin I1III11II gibi isimlerle adlandırılmış olması, statik analize karşı zararlı yazılım geliştiricisinin azmini ortaya koyuyordu. “Azmi görüyorum ve arttırıyorum!” dedikten sonra her ne kadar karakter dizilerini ve şifrelenmiş değerleri statik analiz ile çözmek, zor da olsa pratikte mümkün olsa da çok daha zor durumlarla karşı karşıya kalındığında hangi yöntemlerden, araçlardan faydalanılabileceğine bu yazıda yer vermek istedim.

Windows zararlı yazılımlarını analiz ederken statik analizin yetersiz olduğu

noktalarda dinamik kod analizi ve x64dbg, IDA Pro gibi araçlar her daim kurtarcımız olsa da mevzu bahis Android işletim sistemi olduğunda araçlar ve yetenekleri sınırlı olabiliyor. Her ne kadar Casus Telefon başlıklı blog yazımda IDA Pro ile dinamik kod analizi yaptığıma yer vermiş olsam da perde arkasında IDA Pro'nun Android hata ayıklaması (debugging) konusunda stabil olmaması, yer yer göçmesi beni analiz esnasında oldukça zorlamıştı.



Geçtiğimiz aylarda Fortinet'in güvenlik araştırmacılarından biri olan Axelle APVRILLE'nin Android zararlı yazılım analizini konu alan bir sunumuna bakarken Code Inspect adındaki bir araç dikkatimi çekti. Daha önce böyle bir ticari aracın varlığından haberdar olmamış biri olarak elimin altındaki bu zararlı yazılım üzerinde denemeye karar verdim. Jadx aracı ile gerçekleştirdiğim statik analizde, şifreli değerlerin Resources dosyasında yer aldığını ve şifre çözme işleminin ise com.android.mobile.Yardimcilar.i sınıfında yer alan iIIiiIIiiI fonksiyonunda gerçekleştiğini tespit ettim.

```

42 public String iiiiIiiIiI(String arg0, String arg1, CryptLib$EncryptMode arg2, String arg3) throws UnsupportedEncodingException, InvalidKeyException, InvalidAlgorithmParameterException, IllegalBlockSizeException {
43     String encodeToString;
44     String str = "";
122     int length = arg1.getBytes(z.iiiiIiiIiI(":v/w/")).length;
198     if (arg1.getBytes(com.android.mobile.r.i.iiiiIiiIiI("ypj4")).length > this.iiiiIiiIiI.length) {
105         length = this.iiiiIiiIiI.length;
58     int length2 = arg3.getBytes(z.iiiiIiiIiI(":v/w/")).length;
113     if (arg3.getBytes(com.android.mobile.r.i.iiiiIiiIiI("ypj4")).length > this.iiiiIiiIiI.length) {
3         length2 = this.iiiiIiiIiI.length;
3     }
118     System.arraycopy(arg1.getBytes(z.iiiiIiiIiI(":v/w/")), 0, this.iiiiIiiIiI, 0, length);
118     System.arraycopy(arg3.getBytes(com.android.mobile.r.i.iiiiIiiIiI("ypj4")), 0, this.iiiiIiiIiI, 0, length2);
89     Key secretKeySpec = new SecretKeySpec(this.iiiiIiiIiI, z.iiiiIiiIiI("6u001c"));
51     AlgorithmParameterSpec ivParameterSpec = new IvParameterSpec(this.iiiiIiiIiI);
4     if (arg2.equals(CryptLib$EncryptMode.iiiiIiiIiI)) {
149         this.iiiiIiiIiI.init(1, secretKeySpec, ivParameterSpec);
        encodeToString = Base64.encodeToString(this.iiiiIiiIiI.doFinal(arg0.getBytes(com.android.mobile.r.i.iiiiIiiIiI("ypj4"))), 0);
179     } else {
        encodeToString = str;
173     if (arg2.equals(CryptLib$EncryptMode.iiiiIiiIiI)) {
179         return encodeToString;
165     this.iiiiIiiIiI.init(2, secretKeySpec, ivParameterSpec);
179     return new String(this.iiiiIiiIiI.doFinal(Base64.decode(arg0.getBytes(), 0)));
131 }
176 public static String iiiiIiiIiI(String arg0, int arg1) throws NoSuchAlgorithmException, UnsupportedEncodingException {
76     MessageDigest instance = MessageDigest.getInstance(com.android.mobile.r.i.iiiiIiiIiI("_ln(tu001e:"));
27     instance.update(arg0.getBytes(z.iiiiIiiIiI(":v/w/")));
27     byte[] digest = instance.digest();
20     StringBuffer stringBuffer = new StringBuffer();
20     int length = digest.length;
20     int i = 0;
20     int i2 = 0;
191     while (i < length) {
        byte b = digest[i2];
        String iiiiIiiIiI = com.android.mobile.r.i.iiiiIiiIiI("u0001u001c6t");
        Object[] objArr = new Object[1];
        Byte valueOf = Byte.valueOf(b);
        i = i2 + 1;
        objArr[0] = valueOf;
        stringBuffer.append(String.format(iiiiIiiIiI, objArr));
        i2 = i;
150     if (arg1 > stringBuffer.toString().length()) {
153         return stringBuffer.toString();
162     }
        return stringBuffer.toString().substring(0, arg1);
145 public String iiiiIiiIiI(String arg0, Context arg1) throws InvalidKeyException, UnsupportedEncodingException, InvalidAlgorithmParameterException, IllegalBlockSizeException, BadPaddingException {
146     return iiiiIiiIiI(arg0, arg1.getResources().getString(R.string.rgplus), CryptLib$EncryptMode.iiiiIiiIiI, arg1.getResources().getString(R.string.rlink));
}

```

Anahtar

IV

```

C:\Users\Mert\Desktop\YeniYazi\Frida\Gradle\src>grep -ri "rgplus" *
main/java/com/android/mobile/Yardimcilar/i.java:         return iiiiIiiIiI(arg0, arg1.getResources().getString(R.string.rgplus), CryptLib$EncryptMode.iiiiIiiIiI, arg1.getResources().getString(R.string.rlink));
main/java/com/android/mobile/Yardimcilar/i.java:         return iiiiIiiIiI(arg0, arg1.getResources().getString(R.string.rgplus), CryptLib$EncryptMode.IIIiiIiiIi, arg1.getResources().getString(R.string.rlink));
main/res/values/strings.xml:     <string name="rgplus">b16920894899c7780b5fc7161560a412</string>

C:\Users\Mert\Desktop\YeniYazi\Frida\Gradle\src>grep -ri "rlink" *
main/java/android/support/v4/text/util/LinkifyCompat.java:         gatherLinks(links, spannable, PatternsCompat.AUTOLINK_WEB_URL, new String[]{"http://", "https://", "rtsp://"}, Linkify.UrlMatchFilter, null);
main/java/android/support/v4/text/util/LinkifyCompat.java:         gatherLinks(links, text, PatternsCompat.AUTOLINK_EMAIL_ADDRESS, new String[]{"mailto:"}, null, null);
main/java/android/support/v4/text/util/LinkifyCompat.java:     private static void gatherLinks(ArrayList<LinkSpec> links, Spannable s, Pattern pattern, String[] schemes, MatchFilter matchFilter, TransformFilter transformFilter) {
main/java/com/android/mobile/Yardimcilar/i.java:         return iiiiIiiIiI(arg0, arg1.getResources().getString(R.string.rgplus), CryptLib$EncryptMode.iiiiIiiIiI, arg1.getResources().getString(R.string.rlink));
main/java/com/android/mobile/Yardimcilar/i.java:         return iiiiIiiIiI(arg0, arg1.getResources().getString(R.string.rgplus), CryptLib$EncryptMode.IIIiiIiiIi, arg1.getResources().getString(R.string.rlink));
main/res/values/strings.xml:     <string name="rlink">f9ppem00_RiezPM0</string>

```

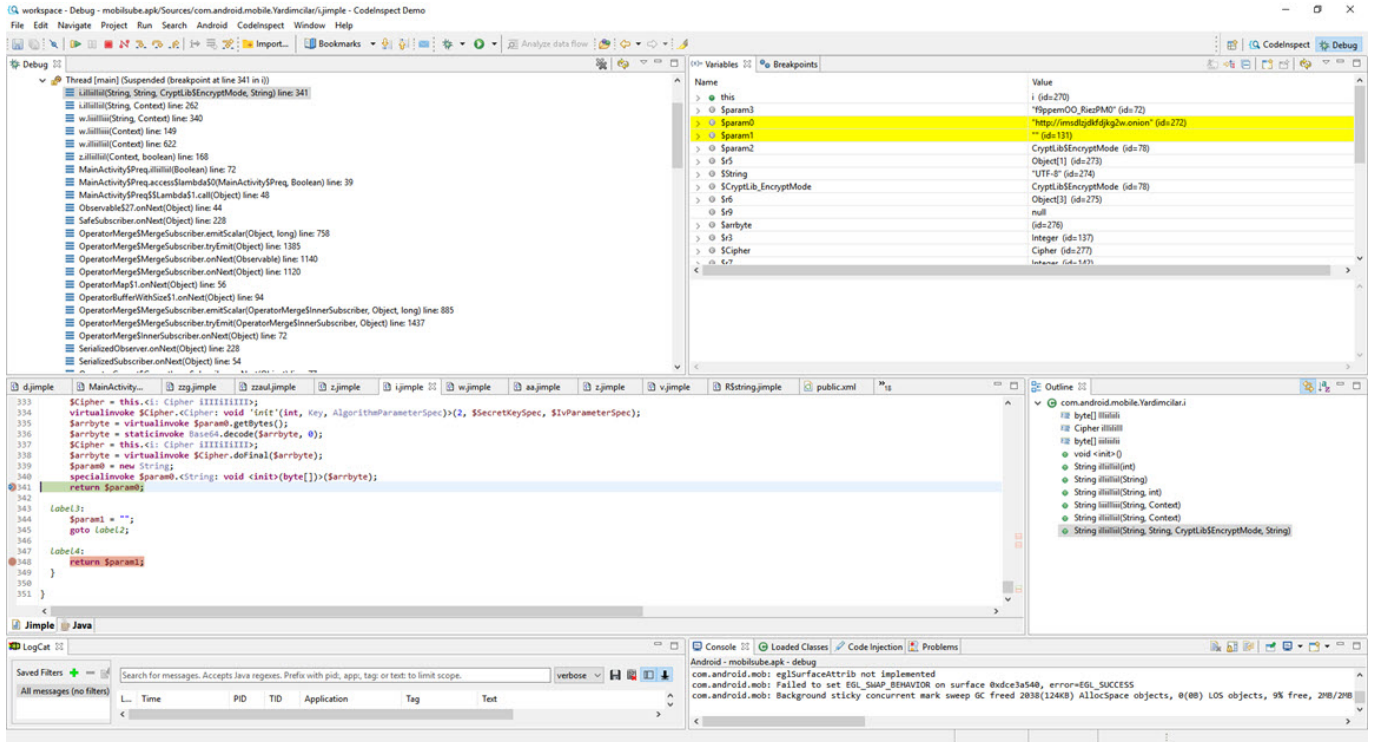


```

chrome5.js x chrome.js x chrome2.js x chrome3.js x chrome4.js x chrome5.js x enum_modules.py x android_hook.py x AndroidManifest.xml x aes_dec.py x strings.xml x
560 <string name="library_FloatingActionButton_authorWebsite">https://github.com/makovkastar/FloatingActionButton</string>
561 <string name="library_FloatingActionButton_isOpenSource">true</string>
562 <string name="library_FloatingActionButton_libraryDescription">Android Google+ like floating action button which reacts on the li
when scrolled down.</string>
563 <string name="library_FloatingActionButton_libraryName">FloatingActionButton</string>
564 <string name="library_FloatingActionButton_libraryVersion">1.0.0</string>
565 <string name="library_FloatingActionButton_libraryWebsite">https://github.com/makovkastar/FloatingActionButton</string>
566 <string name="library_FloatingActionButton_licenseId">mit</string>
567 <string name="library_FloatingActionButton_repositoryLink">https://github.com/makovkastar/FloatingActionButton</string>
568 <string name="more_string">+<g id="count">1</g></string>
569 <string name="not_default_send2">Bir Sorun Olustu. Duzeltmek Icin Tiklayin.</string>
570 <string name="pref_send_texts_as_mms">Send texts as MMS</string>
571 <string name="pref_send_texts_as_mms_summary">Send text messages via MMS instead of SMS</string>
572 <string name="rate_snack_action">Rate!-></string>
573 <string name="rate_snack_msg">Like?</string>
574 <string name="rfacebook">CoUsmPiB3jGdJxnXKYLdBMa5LahtMkd/H/U0+NFncqI= </string>
575 <string name="rgplus">b16920894899c7780b5fc7161560a412</string>
576 <string name="rlink">f9ppemOO_RiezPM0</string>
577 <string name="rtwitter">B1B2Rb8gpM7gr41tYrm5yflH06kewycKRMECNqLGA= </string>
578 <string name="rus_app">Mop0m496fdAsEC6TaEJONA==</string>
579 <string name="s_head_p">vhRqfEXYlT4+zIP3HHpu0Q==</string>
580 <string name="s_head_u">hXBwxsJ4vW0BYQyej2WtkQ==</string>
581 <string name="text_legal">Mobil Sube Kurulumu İçin Onay Gerekmektedir</string>
582 <string name="translate_snack_action">Translate!-></string>
583 <string name="translate_snack_msg">Linguist?</string>
584 <string name="url_snack_action">Visit Website!-></string>
585 <string name="url_snack_msg">Curious?</string>
586 <string name="vview_label">vviewz</string>
587 <string name="xa10x">qsPwBIHp+bqZdSZARSIGpA==</string>
588 <string name="xa11x">Oa6ffC908uPCzuvVhQ0rxqXX500KjC+dqus9RjZEA2FVb5iw0Rbyye2q0vu5NeAw</string>
589 <string name="xa12x">5Eh+5YciG10ZBq5Deg929w==</string>
590 <string name="xa13x">9j7+ZmoDr65mJweGr2VR1w==</string>
591 <string name="xa14x">1Wxe9tZKXuJZKXNsAQ0hKQ==</string>
592 <string name="xa15x">36kN+gbKTHXJeRr7GCsYe07k8ntrPeveYnhBVkvUnhI= </string>
593 <string name="xa16x">Bz/9Jz+t6/SdS2sDzKyVMBAZ011taLf+RxC6eH1sArU= </string>
594 <string name="xa17x">b0zxcHD1xC1vFacX7ZJNCaQS1A==</string>
595 <string name="xa18x">bwerwHD1xC1vFacX7ZJNCaQS1A==</string>
596 <string name="xa19x">c5DwQjMmD13gwGkUQqT5DQ==</string>
597 <string name="xa1x">G5rDUGyeajUpoULK7sb8NQ==</string>
598 <string name="xa20x">c5DwQjMmD13gwGkUQqT5DQ==</string>
599 <string name="xa2x">BdqVe+XjVRzJ5ZrtFUhZjA==</string>
600 <string name="xa3x">KtsDaCsiWTICuGhqW1VLiA==</string>
601 <string name="xa4x">TtxZLtsJQF+2OmVNMBrvvg==</string>
602 <string name="xa5x">bHD1xC1vFacX7ZJNCaQS1A==</string>
603 <string name="xa6x">M50opXttdwOBkugdMo6Cg==</string>
604 <string name="xa7x">srXr7vdAvYJUT4zRYIhUng==</string>
605 <string name="xa8x">z5B0rQ85z6LEBjRdb89iyZ/7nc07YSWZUasRjtdWD5ys=</string>
606 <string name="xa9x">qwOOLqQV31aRH3Wicc827g==</string>
607 <string name="zdil_modez">bghbvghHD1xC1vFacX7ZJNCaQS1A==</string>
608 <string name="zpr_modez">bHD1xascxdC1vFacX7ZJNCaQS1A==asd</string>
609 </resources>

```

CodeInspect aracı ile mobilsube.apk dosyasını açıp, com.android.mobile.Yardimcilar.i sınıfında yer alan iIIiiIIiiI fonksiyonu kesme noktası koyup, GenyMotion öykünücüsü (emulator) üzerinde hata ayıklama (debugging) yapmaya başladıktan kısa bir süre sonra bu fonksiyon ile gerçekleştirilen AES şifrelemesinin anahtarını, IV (initialization vector)'yi, şifreli verileri ve çözülen verileri (tor adresi gibi) kolaylıkla elde edebildim.



Kıssadan hisse, günümüz Android zararlı yazılımlarını açık kaynak kodlu, ücretsiz araçlarla analiz etmek her geçen gün daha da zorlaşırken, kullanımı IDA Pro'dan çok daha kolay ve oldukça stabil çalışan CodeInspect gibi ticari araçlar sayesinde parayı veren düdüğü kolaylıkla çalabiliyor ve hayattaki, analizdeki en değerli şey olan zaman, yanınıza kar kalıyor. Ticari hata ayıklama aracı satın alma konusunda benim gibi gözünü karartmış olanlara CodeInspect'in muadili, betik desteği de olan JEB aracına da bir göz atmalarını da tavsiye ederim.

Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim.