Analisis dokumen berbahaya – Bagian 02 – Dokumen PDF

Setelah tinjauan tentang pengelolaan mesin virtual dan instalasi Remnux sebagai sistem operasi untuk menganalisis artefak yang mencurigakan, kita akan menjelajahi berkas PDF dalam hal format dan bagaimana ia digunakan untuk merugikan pengguna.

Apa itu PDF?

Saat kita membuka berkas PDF, kita menggunakan perangkat lunak khusus yang membuat kontennya mudah dibaca, namun, seperti banyak jenis berkas lainnya, PDF dibuat menggunakan kombinasi teks biasa dan data biner (untuk gambar dan elemen lain yang mungkin memerlukannya), misalnya, teks berikut menampilkan berkas PDF yang ditampilkan setelahnya:

```
%PDF-1.4
1 0 obj
<<
        /Panjang 51
>>
aliran
1 0 0 RG
5 w
36. 144 m
180 144 l
180 36 l
36 36 l
s
endstream
endobj
2 0 obj
<<
        /Jenis /Katalog
        /Halaman 3 0 R
>>
endobj
3 0 obj
```

```
<<
        /Jennis /Halaman
        /Anak-anak [4 0 R ]
        /Hitung 1
>>
endobj
4 0 obj
<<
        /Jenis /Halaman
        /Induk 3 0 R
        /Kotak Media [0 0 612 792]
        /Isi 1 0 R
>>
endobj


xref
0 4
0000000000 65535 f
0000000010 00000 n
0000000113 00000 n
0000000165 00000 n
0000000227 00000 n
cuplikan
<<
        /Ukuran 4
        /Root 2 0 R
>>
startxref
344
%%EOF
```

```
<img src="https://greaterinternetfreedom.org/wp-
content/uploads/2023/07/example.png">
```

Dari “Cara membuat berkas PDF sederhana” oleh Callas Software

Struktur berkas

Dari contoh ini, kita dapat melihat struktur standar berkas PDF apa pun:

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/example2.png">
```

Header: Berisi versi protokol yang digunakan untuk membuat berkas, untuk menginstruksikan program pembaca cara membaca sisa struktur dan merender semua elemennya.

Isi: Di sini akan terdapat semua objek yang menyusun berkas PDF, halaman, gambar, teks, font, dll. Bahkan kode dan tindakan otomatis jika bekas tersebut memilikinya.

Tabel referensi silang: Di sini kita akan menemukan daftar semua objek dokumen untuk memudahkan akses dan tempatnya masing-masing di dalam file. Ini mirip dengan 'daftar isi' tetapi untuk dibaca oleh perangkat lunak pembaca PDF. Jika pada waktu tertentu pembaca perlu merender objek tertentu (misalnya, jika kita menggulir ke halaman acak pada dokumen besar), perangkat lunak pembaca akan melihat halaman mana yang harus dirender, dan mencarinya di tabel ini untuk menemukan lokasi masing-masing elemen di bagian isi untuk memuatnya di layar.

Cuplikan: Di sini kita akan menemukan tempat dalam dokumen tabel referensi silang dan informasi berguna lainnya, seperti jumlah objek dalam tabel referensi silang (misalnya untuk memeriksa bahwa file tidak rusak), objek root dari dokumen, dan informasi enkripsi jika tersedia. Pembaca PDF dengan sengaja mulai membaca dokumen dari akhir, di mana mereka dapat dengan cepat menemukan lokasi objek root, dan tabel referensi silang untuk mulai merender konten.

Memahami objek PDF

Mengingat cara PDF disusun, bagian yang paling menarik untuk dipelajari lebih dalam adalah isi, karena di sana kita dapat menemukan segala sesuatu yang dapat kita lihat dan berinteraksi (teks, halaman, gambar, formulir, kode, dll.). Semua elemen ini disebut objek, dan bermacam-macam elemennya sesuai dengan spesifikasinya, misalnya pada contoh di atas, objek tersebut

```
4 0 obj
<<
        /Jenis /Halaman
```

```
        /Induk 3 0 R
        /Kotak Media [0 0 612 792]
        /Isi 1 0 R
>>
endobj
```

adalah halaman dengan id 4, memiliki elemen induk dengan id 3, dan berisi elemen anak dengan id 1 (persegi panjang merah). Menampilkan cara persis setiap objek ditulis berada di luar cakupan materi ini, namun, ada beberapa jenis elemen yang mungkin digunakan untuk tujuan jahat, dan kami ingin membahasnya lebih jauh.

/OpenAction: Objek ini mereferensikan serangkaian tindakan yang akan dijalankan ketika berkas PDF dibuka, ini mungkin digunakan untuk membuka situs web untuk dukungan konten atau untuk tujuan pelacakan, menjalankan kode JavaScript, dll. Ini mungkin digunakan untuk mengelabui pengguna untuk melakukan tindakan tambahan yang dapat membahayakan, atau bergantung pada sistem operasi komputer, objek ini dapat menjalankan malware secara langsung.

/AA: Objek ini juga mencakup serangkaian tindakan yang dipicu dalam berbagai keadaan, seperti memvisualisasikan halaman, mengarahkan penunjuk tetikus ke objek tertentu, mengisi kolom formulir, dll. Risiko terkait serupa dengan objek /OpenAction, tetapi dengan masih banyak lagi skenario pemicu.

/JS atau /Javascript: Berisi kode JavaScript yang akan dijalankan setelah suatu tindakan dipicu, kode ini mungkin menyertakan fungsi eksklusif untuk PDF.

/Peluncuran: Mencoba meluncurkan aplikasi eksternal pada perangkat setelah suatu tindakan dipicu, ini mungkin digunakan, misalnya, untuk membuka dokumen lain atau menjalankan perintah tertentu, yang mungkin berbahaya.

/EmbeddedFile: Memungkinkan penyertaan berkas arbitrer, dari dokumen hingga berkas yang dapat dijalankani di dalam berkas PDF. Ada sebuah berkas yang mendahui berkas PDF biasa yang ditanamkan berkas berbahaya lainnya, seperti malware yang dapat dijalankan atau dokumen Microsoft Office dengan makro berbahaya.

/ObjStm: Berisi informasi arbitrer yang akan diproses sesuai dengan cara pemanggilannya. Kegunaan utama objek ini adalah untuk mengelompokkan banyak objek dan mengompresinya sehingga menjadi berkas yang lebih kecil. Namun, ini juga dapat digunakan untuk mengompresi kode berbahaya sebagai cara untuk

mengaburkannya dan menghindari deteksi antivirus. Mengingat banyaknya kasus penggunaan yang berbeda untuk jenis objek ini, mengasumsikan keberadaannya sebagai objek berbahaya akan membawa kita pada banyak kesalahan positif.

Dalam serangan yang lebih rumit, kombinasi objek-objek ini dapat digunakan, misalnya, sebuah berkas PDF bisa memicu tindakan membuka berkas yang tertanam di dalam berkas yang sama yang dienkripsi di /ObjStm.

Mempertimbangkan semua ini, kami ingin mengetahui apakah suatu berkas memiliki salah satu tipe objek ini sebagai langkah pertama untuk melihat apakah berkas PDF berbahaya, atau setidaknya untuk memastikan bahwa ia tidak berbahaya.

Masukkan pdfid

Mengingat kita tahu harus mulai mencari tanda bahaya pada berkas PDF yang mungkin kita anggap mencurigakan, kita bisa mulai menggunakan alat pdfid sebagai langkah pertama untuk melihat tipe objek mana yang terdapat dalam berkas kita. Pdfid adalah bagian dari rangkaian alat yang dikembangkan oleh Didier Stevens untuk menyederhanakan beberapa proses analisis pada berkas PDF. Alat-alat ini dijalankan menggunakan baris perintah, sehingga dikenal sebagai aplikasi CLI (Command Line Interface). Kami akan menjelaskan cara menggunakannya dengan Mesin Virtual Remnux yang kami siapkan di bagian sebelumnya dari kursus ini.

Untuk menggunakan pdfid, kita perlu membuka aplikasi Terminal di mesin virtual kita. Saat kita memulai VM, jendela ini seharusnya sudah terbuka, namun kita selalu dapat mengklik menu Aktivitas di pojok kiri atas, lalu ikon Terminal di panel kiri, seperti yang ditunjukkan pada gambar,

Setelah berada di jendela Terminal, kita bisa mulai mengeksplorasi penggunaan perintah pdfid melalui perintah untuk menampilkan bantuannya, kita hanya perlu mengetik:

```
pdfid.py -h
```

-h untuk “**b**antuan”

Tip: kita selalu dapat menggunakan tombol Tab untuk melengkapi beberapa perintah secara otomatis

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/Screenshot-from-2022-06-13-15-32-42.png">
```

Di sini kita dapat melihat sejumlah opsi yang dapat kita terapkan saat menggunakan pdfid, ini mungkin terlihat menantang bagi mereka yang baru menggunakan terminal, namun, kami biasanya tetap menggunakan beberapa opsi ini, dan juga dengan sedikit latihan, prosesnya menjadi lebih cepat dan lebih mudah.

Untuk menganalisis file pertama kita, kita perlu memastikan bahwa perintah yang kita jalankan di baris perintah dijalankan “dari folder direktori”, jadi kita perlu mengetahui dari mana perintah tersebut dijalankan, dan di mana letak berkas yang kita inginkan. untuk dianalisa. Untuk memberi Anda beberapa konteks, setiap kali kita membuka aplikasi Terminal di Remnux, buka terminal di direktori Home, lokasi yang sama yang kita lihat ketika kita membuka aplikasi Berkas

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/files-icon.png">
```

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/files.png">
```

Untuk mempermudah, kita dapat meletakkan PDF kita di folder ini, sehingga perintah terminal dijalankan dari direktori yang sama dengan berkas tersebut.

Kita dapat mengambil contoh file kita dari atas dan menyimpannya sebagai berkas pdf dengan bantuan editor teks di komputer host kita dan seret dan letakkan berkas tersebut ke direktori home Remnux.

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/Screenshot-from-2022-06-13-16-02-30.png">
```

Dan dengan ini, kita dapat menjalankan perintah berikut di Terminal:

```
pdfid.py tes.pdf
```

Untuk menerima tanggapan ini:

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/Screenshot-from-2022-06-13-16-04-20.png">
```

Seperti yang dapat kita periksa, semua objek yang dilihat oleh pdfid cocok dengan yang kita ketahui dari sumber berkas PDF, dan sepertinya tidak ada satupun yang masuk dalam daftar objek mencurigakan yang telah dijelaskan di atas.

Seperti yang kami nyatakan sebelumnya, ada teknik yang digunakan pelaku jahat untuk menghindari deteksi objek tertentu dengan mudah, pdfid mencoba menampilkan objek yang bahkan dikaburkan, namun, dalam beberapa kasus yang kurang umum, mungkin ada objek tersembunyi yang memerlukan penelusuran lebih dalam untuk menemukannya.

Masukkan pdf-parser, contoh 1

Sekarang, apa yang terjadi jika kita menemukan berkas pdf dengan objek mencurigakan? Bayangkan kita memiliki berkas ex005.pdf yang memberi kita output seperti ini di pdfid:

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/ex005_1.png">
```

Dari sini, dan panduan sumber daya di atas, kita mengetahui bahwa ada 3 objek bertipe /JS, /Javascript, dan /OpenAction yang mungkin menarik untuk diulas, terutama karena objek tersebut menunjukkan bahwar berkas tersebut mencoba menjalankan suatu tindakan saat kita membukanya. Disini kita bisa memprosesnya dengan pdf-parser untuk mengetahui tipe masing-masing objek dan melihat isinya. Untuk berkas contoh, kita akan menjalankan perintah berikut:

```
pdf-parser.py -a ex005.pdf
```

Kita menggunakan argumen -a untuk melihat statistik berkas, untuk acuan yang lebih baik kita selalu dapat menggunakan pdf-parser.py –help untuk melihat daftar pilihan di layarnya.

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/ex005_2.png">
```

Di sini kita dapat melihat bahwa memang kita memiliki ketiga objek bermasalah tersebut, tetapi juga beberapa informasi tambahan, kita dapat melihat id objek untuk setiap jenis objek. Sekarang kita tahu bahwa objek /JS dan /Javascript berada di objek dengan id 7, dan objek /OpenAction berada di objek dengan id 1. Selanjutnya, kita dapat melihat konten objek /OpenAction ke lihat apa yang coba dilakukan dokumen ini saat dibuka, untuk ini kita menggunakan perintah

```
Pdf-parser.py -o 1 ex005.pdf
```

Di sini argumen -o digunakan untuk memberi alat itu id objek yang kontennya ingin kita lihat di layar:

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/ex005_3.png">
```

Di sini kita dapat melihat baris “/OpenAction 7 0 R”, yang berartikonten sebenarnya dari objek /OpenAction berada di objek dengan id 7, dan ketika kita membuka berkas tersebut, kita akan memanggil atau mengacu pada objek tersebut. Mengulangi proses tadi untuk melihat isi objek dengan id 7 kita mendapatkan:

Di mana kita dapat melihat bahwa dokumen tersebut mencoba menampilkan peringatan atau pop-up dengan pesan yang dijelaskan di terminal, jika kita membuka dokumen tersebut akan terlihat seperti ini:

Contoh 2

Seperti yang kami sebutkan sebelumnya, mungkin ada file yang konten mencurigakannya tidak dapat dilihat dalam teks biasa, mungkin ada sejumlah alasan untuk melakukan hal ini untuk hal yang sah, seperti mengompresi informasi yang panjang untuk mengurangi ukuran berkasnya, dan lain-lain, namun, berkas berbahaya menggunakan teknik ini dengan tujuan untuk membuat bingung dan membantu menghindari deteksi oleh perangkat lunak antivirus dan solusi keamanan lainnya. Misalnya, jika kita mengulangi alur kerja sebelumnya pada berkas ex006.pdf, kita akan melihat bahwa output untuk perintah pdfid adalah sebagai berikut:

Di sini kita bisa melihat pada baris /JavaScript “1(1)”, artinya pdfid mendeteksi objek dengan jenis ini, namun dikaburkan, dengan mengulangi alur kerja yang sama yang sudah kita ketahui, kita tinjau objek tersebut dengan id 8 (di mana kode JavaScriptnya berada) untuk melihat hal berikut:

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/ex006_2.png">
```

Di sini kita tidak dapat melihat kode sebenarnya seperti pada contoh terakhir, sebaliknya kita melihat antara lain baris “/Filter /FlateDecode”. Opsi /Filter menjalankan tindakan pada konten akhir aliran untuk memecahkan kodenya, kemudian /FlateDecode menunjukkan pengkodean terkait yang harus dipertimbangkan ketika mendekode konten, untuk memberikan pemahaman yang lebih baik tentang hal ini, jika kita membuka berkas dengan teks editor dan cari elemen ini secara manual, kita akan melihat sesuatu seperti ini:

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/ex006_3.png">
```

Di mana konten di dalam kotak merah adalah konten yang dikodekan sebenarnya, untuk kasus ini, parser pdf dapat mencoba memecahkan kode konten sebenarnya, untuk ini kita mengunakan argumen -f, jadi kita menggunakan perintah

```
Pdf-parser.py -o 8 -f ex006.pdf
```

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/ex006_4.png">
```

Dimana kita bisa melihat konten sebenarnya dari objek yang akan direnderer oleh PDF reader.

Sekarang setelah kita mengetahui dasar-dasar cara meninjau berkas PDF untuk mencari objek mencurigakan, ada beberapa tantangan untuk Anda.

Contoh 3

Hal lain yang biasanya dilakukan perangkat lunak pembuat PDF untuk membuat berkas

baru adalah membuat objek di dalam aliran yang dikodekan untuk membuat berkas yang dihasilkan menjadi lebih kecil, hal ini diinginkan secara umum, tetapi juga menciptakan cara untuk lebih mengaburkan kode berbahaya, menganalisis berkas example3.pdf memperlihatkan kita beberapa /ObjStm (Object Streams) yang mungkin berisi (dan memang demikian) objek lain yang mungkin menarik.

Untuk skenario seperti ini, disarankan untuk menggunakan opsi -O (seperti dalam huruf kapital o) dari parser pdf, opsi ini akan mencoba mengurai aliran apa pun yang berisi objek dan memperlakukannya sebagai objek dari berkas biasa, misalnya, menggunakan opsi ini seperti berikut.

Mengungkapkan bahwa berkas tersebut memiliki objek "baru" dan salah satunya adalah/AA, yang membuatnya menarik untuk melakukan pencarian atas perilaku berbahaya, dengan melihat masing-masing objek yang kita miliki

Memberitahu kita bahwa tindakan tersebut terkait dengan objek halaman, dalam hal ini /O menunjukkan bahwa tindakan tersebut dipicu ketika kita membuka halaman, dan tindakan sebenarnya disimpan di objek 37, lalu

Setelah beberapa penelitian, kita dapat menyimpulkan bahwa objek ini mencoba

membuka dialog properti pembaca pdf seperti ini (tidak ada yang berbahaya – contoh di komputer yang dikonfigurasi dalam bahasa Spanyol)

```
<img src="https://greaterinternetfreedom.org/wp-content/uploads/2023/07/example3_5.png">
```

Kesimpulan: coba gunakan parameter -O jika ada objek lain yang bersembunyi di dalam aliran

Tantangan

Pertanyaan: menganalisis berkas tantangan1.pdf (md5: 3b20821cb817e40e088d9583e8699938), objek menarik apa yang bersembunyi di balik sebuah aliran?

1. TindakanTerbuka
Salah – …

2. AA
Benar - …

3. JS
Salah – …

4. JavaScript
Salah – …

Pertanyaan: menganalisis berkas tangtangan2.pdf (md5: 30373b268d516845751c10dc2b579c97), kita dapat melihat tindakan yang mencoba membuka URL, kode apa yang disertakan dalam URL sebagai kode pelacakan? (petunjuk: 6 angka)

Saya ingin melihat jawabannya
88965

Sekarang setelah kita mengetahui lebih banyak tentang PDF, mari kita bahas di

bagianselanjutnya jenis berkas lain yang sangat sering digunakan: Dokumen Office..