FB

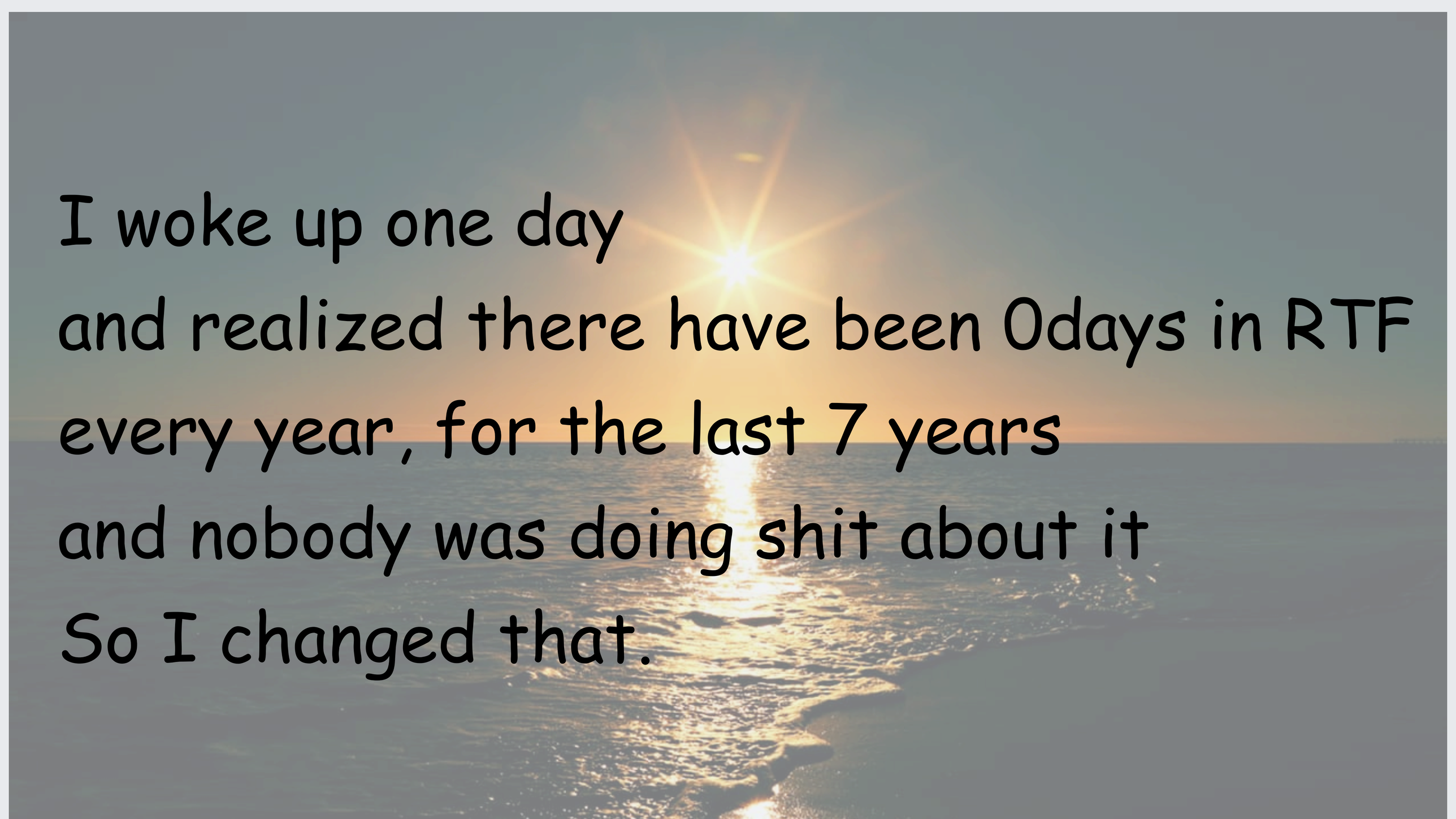
Advanced Analysis of Malicious Documents

Matt Richard

Real h4x0rs dr4g and dr0p 0day

MJR

Bit Policy Directorate


I woke up one day
and realized there have been 0days in RTF
every year, for the last 7 years
and nobody was doing shit about it
So I changed that.

Adobe Flash Zero-Day Under Attack

By Ryan Naraine on May 10, 2016

A zero-day vulnerability in Adobe's ubiquitous Flash Player software is being exploited to launch malware attacks, the company warned in an advisory issued today.

The vulnerability, rated critical, will not be patched until May 12th.

The company credits Genwei Jiang of FireEye, Inc. with discovering the flaw, which provides an indication that it is being used in targeted attacks.

According to Adobe, the vulnerability is present in Windows, Mac OS X, Linux and Chrome OS.

From the **Adobe advisory**:

> A critical vulnerability (CVE-2016-4117) exists in Adobe Flash Player 21.0.0.226 and earlier versions for Windows, Macintosh, Linux, and Chrome OS. Successful exploitation could cause a crash and potentially allow an attacker to take control of the affected system.
>
> Adobe is aware of a report that an exploit for CVE-2016-4117 exists in the wild. Adobe will address this vulnerability in our monthly security update, which will be available as early as May 12.

Once upon a time, in 2009…

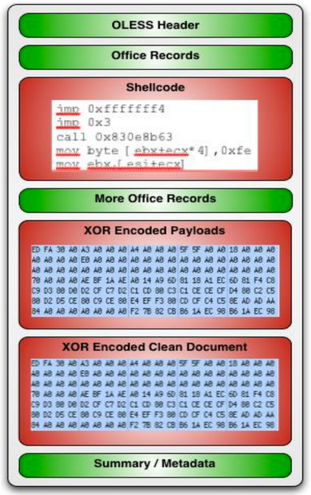
Raytheon
Customer Success Is Our Mission

Advanced Analysis of Malicious Documents

Matt Richard

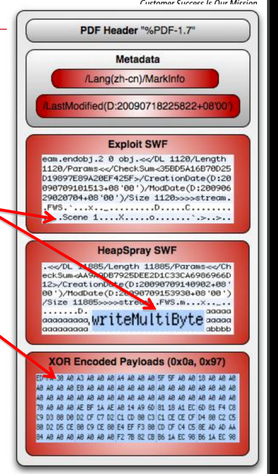
Office Exploit Structure

- Everything included
- Permutations
  - Clean document
  - Trojan payload
  - Shellcode
  - Obfuscations



PDF 0-day 7/15/2009

- Adobe > v9 include Flash
- Flash is a programming language
  - Perfect for implementing heap sprays
- PDF loads heapspray SWF
- PDF loads exploit SWF
- Payload at fixed offset
- Two payloads
- First payload launches clean PDF
- Second payload installs backdoor



Tools - Yara

- http://code.google.com/p/yara-project/
- Classification
- Windows/*NIX
- No data transforms
- Non-linear scan times
- Simple and correlated rules
  - Ascii, binary, regex, wildcards

```
Rule PDF_Flash_Exploit
{
strings:
   $a = "%PDF-1."
   $j = "(pop\\056swf)"
   $k = "(pushpro\\056swf)"
   $b = "(         a.swf)"
condition:
   ($a at 0) and ($j or $k or $b)
}
```

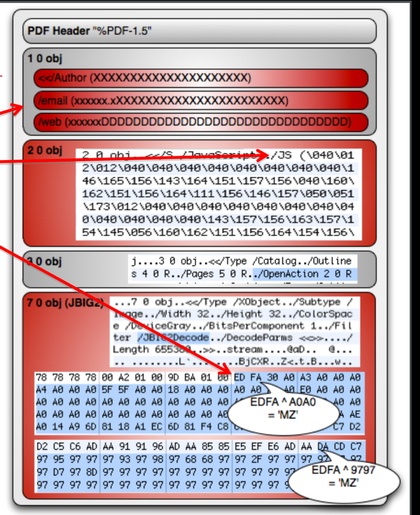
PPT Case Study – Using Yara

```
[matt@localhost]$ yara -s docfiles.yara 1d65f1be33c9d72030508c3df24ec780

EXPERIMENTAL_shellcode_lodsb_xor_stosb_decode
000018EC: EB EA FC 33 C9 EB 01
000018DE: C0 C8 06 75 03 74 01 E8 32 C3 AA 49

HIGH_xor_e
0
```

```
rule EXPERIMENTAL_shellcode_lodsb_xor_stosb_decode
{
strings:
$a = { c0 c8 06 75 03 74 01 ?? 32 c3 aa 49 }
$b = { eb ea fc 33 c9 eb 01 }
condition:
any of them
}
```

JBIG2 PDFs

- First JBIG2 1/15/2009
  - Unique metadata
  - JS Ocal Encoding
  - 1 or 2 Payloads
  - XOR 0xa0 / 0x97



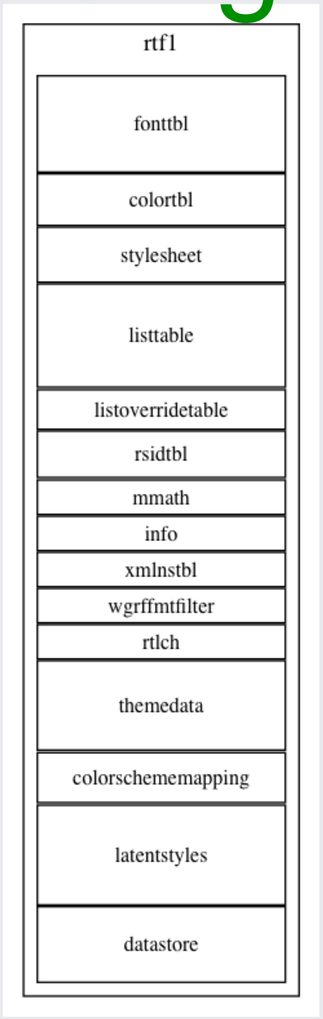
{\rtf1
{\obj}\ansi\deff0{\fonttbl{\f0\fnil\fcharset0 Calibri;}}
{*\generator Msftedit 5.41.21.2510;}\viewkind4\uc1\pard\sa200\sl276\slmult1\lang9\f0\fs22
Reaching a deal on Iran's nuclear program is a matter of political will, and the "security of the world is at stake," the European Union's top diplomat said Sunday.\par
\par
European Union High Representative Federica Mogherini made the comments in Vienna, Austria, ahead of a meeting with U.S. Secretary of State John Kerry.\par
\par
The U.S. and its negotiating partners Britain, France, Germany, Russia and China -- known as the P5+1 -- as well as the European Union hope to reach a comprehensive nuclear deal with Iran before June 30.\par

\par
\par
{\object\objocx\f37\objsetsize\objw1440\objh1440{*\objclass Forms.Image.1}{*\objdata
01050000020000000e000000466f726d732e496d6167652e3100000000000000000000480000
d0cf11e0a1b11ae10000000000000000000000000000000000003e000300feff090006000000000000000000000001000

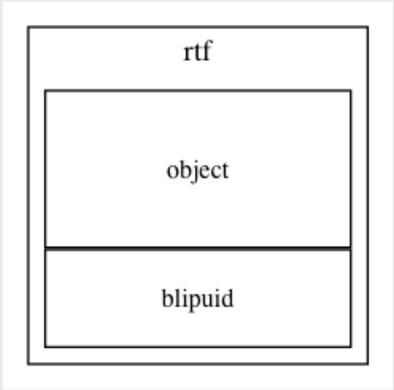
ffffffffffffffffff0000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000105000000000000}}!!!!ñ◆qÕ<NUL>Ò@\9◆BÒ,+ñ◆À◆×F
wà¢◆^U'¶◆¾^A^Z^NxÑº◆Ç"¹ØmU2^DeZS?ê^S[^BpzùÒ¤ú^VÁz:◆Gpì$þeòqY*<FS>I◆<CR>ïné[ªN<FS>9Ý¶<◆gÚ^Aj"ÕL
¼◆×«·mÏ÷0◆Ì◆D9§^N¾Êé°@◆Íæñ◆◆<FS>ú^D,◆°◆◆\e,+vÃ–@\ñ±*>◆@^A%þF^Yû^N¾û◆<BS>AÓ^P×«¿mÏ÷0<NUL>I8µ<ES

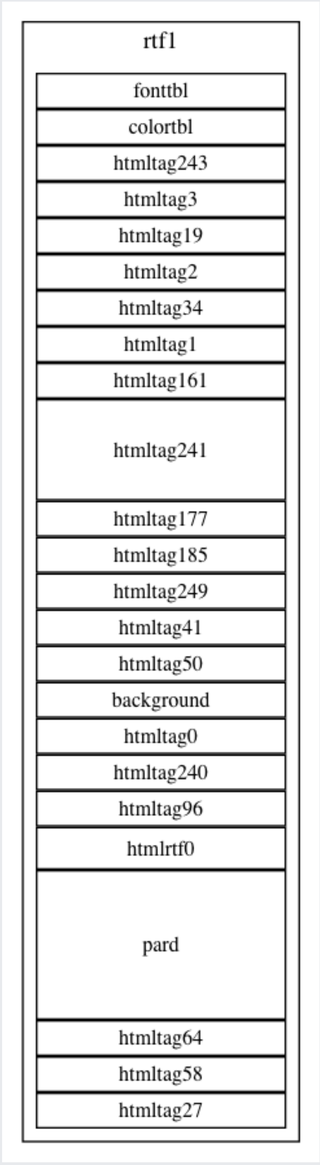
Capture Intuition

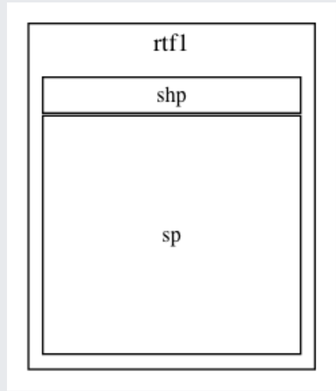
Benign



Malicious





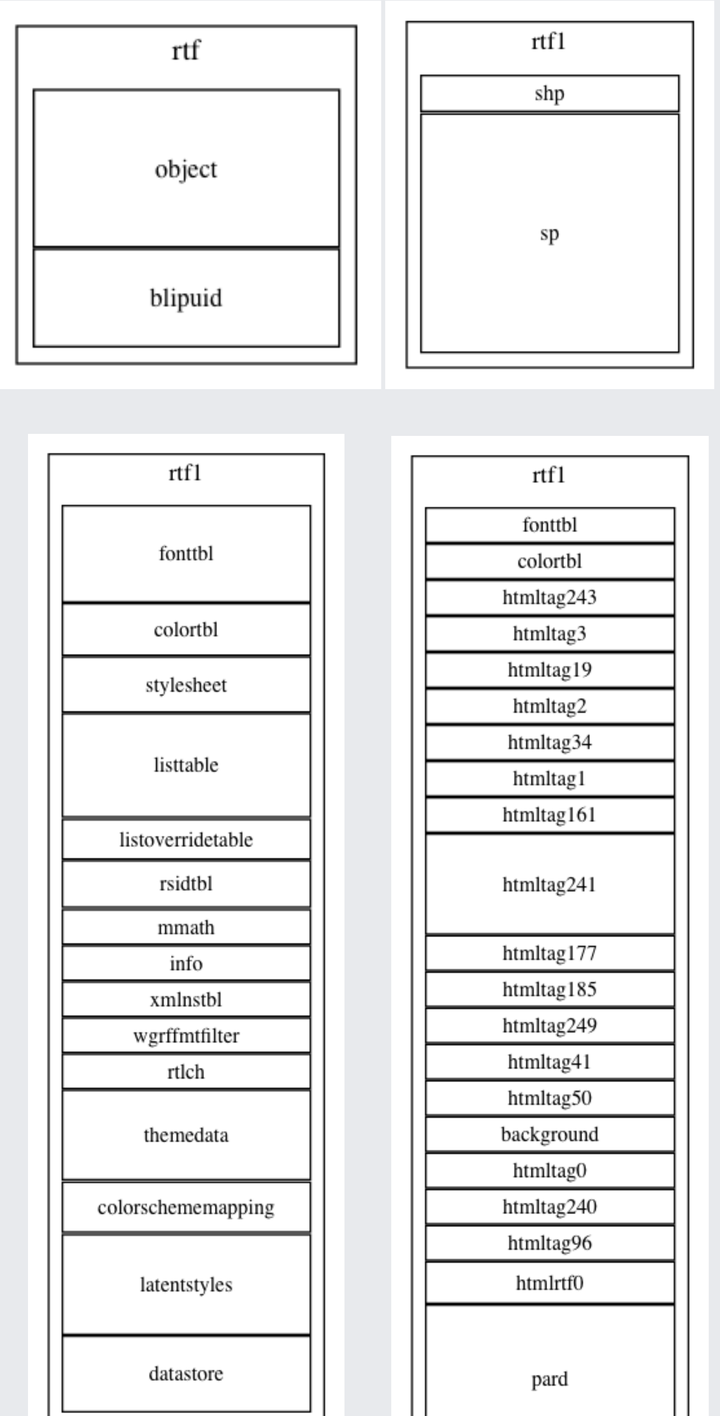


Capture Intuition

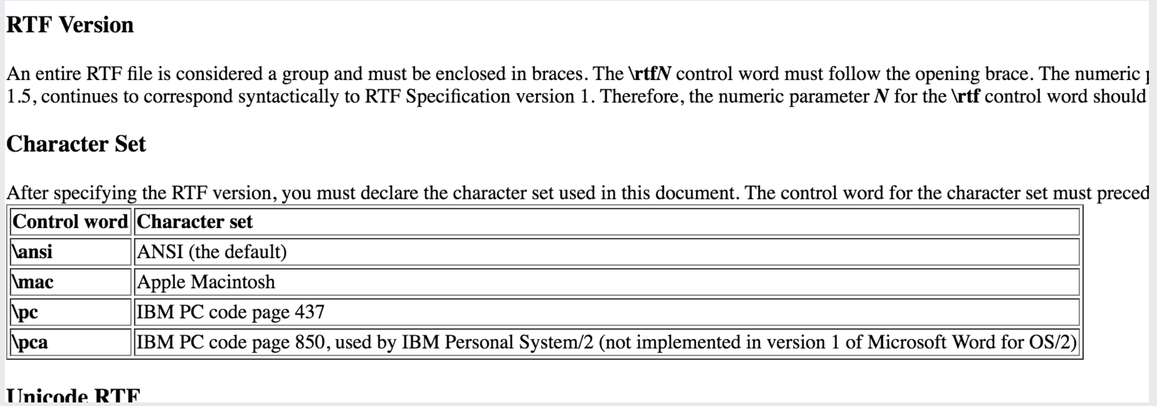
- Hypothesis - to be malicious an RTF:
  - contains content not normally found
  - malicious content changes flow
  - when opened leads to code exec*

- Hypothesis - benign documents are almost exclusively made by tools, the tools create consistent structures

- ***Malicious and non-malicious***



Identify Features



RTF Version

An entire RTF file is considered a group and must be enclosed in braces. The **\rtf*N*** control word must follow the opening brace. The numeric
1.5, continues to correspond syntactically to RTF Specification version 1. Therefore, the numeric parameter ***N*** for the **\rtf** control word should

Character Set

After specifying the RTF version, you must declare the character set used in this document. The control word for the character set must preced

Control word	Character set
\ansi	ANSI (the default)
\mac	Apple Macintosh
\pc	IBM PC code page 437
\pca	IBM PC code page 850, used by IBM Personal System/2 (not implemented in version 1 of Microsoft Word for OS/2)

Unicode RTF

- We theorize that to be malicious certain structures exist
  - objects
  - extra data
  - obfuscation

- Count and measure location of document

Counting Features

```
rule rtf_ole_object
{
meta:
  type = "feature"
strings:
  $a = "{\\*\\datastore"
condition:
  $a
}
```

```
class YaraScan(object):
    results = {}
    data_len = 0

    def __init__(self, rulepath, results_prefix='yara', offset_postfix='offset'):
        self.engine = yara.compile(rulepath)
        self.results_prefix = results_prefix
        self.offset_postfix = offset_postfix

    def offset_ratio(self, strings):
        return min([b[0] for b in strings]) / float(self.data_len)

    def callback(self, data):
        rule = "%s_%s" % (self.results_prefix, data['rule'])
        offset_name = "%s_%s" % (rule, self.offset_postfix)
        if data['matches'] == True:
            match_len = len(data['strings'])
            self.results[rule] = 1 if match_len == 0 else match_len
            self.results[offset_name] = 0 if match_len == 0 else self.offset_ratio(data['strings'])
        else:
            self.results[rule] = 0
            self.results[offset_name] = 0
        yara.CALLBACK_CONTINUE

    def scan_file(self, filename):
        with open(filename, 'rb') as f:
            data = f.read()
        return self.scan_data(data)

    def scan_data(self, data):
        self.data_len = len(data)
        self.reults = {}
        self.engine.match(data=data, callback=self.callback)
        return self.results
```

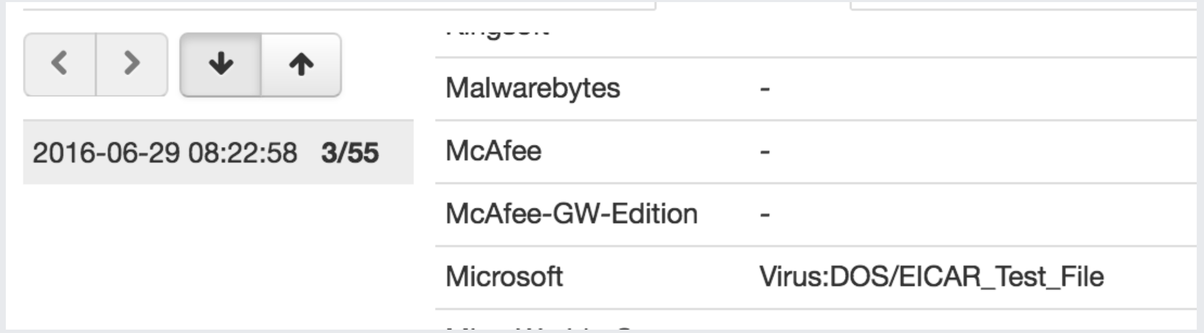
Yara Feature Vectors


```
{'yara_rtf_ancalog': 0,
 'yara_rtf_ancalog_offset': 0,
 'yara_rtf_ansi_char': 3,
 'yara_rtf_ansi_char_offset': 3.8081663407057 62e-05,
 'yara_rtf_author': 1,
 'yara_rtf_author_offset': 0.23472177264702923,
 'yara_rtf_background': 0,
 'yara_rtf_background_offset': 0,
 'yara_rtf_nofeaturethrottle1': 1,
 'yara_rtf_nofeaturethrottle1_offset':
```

Ground Truth Labels

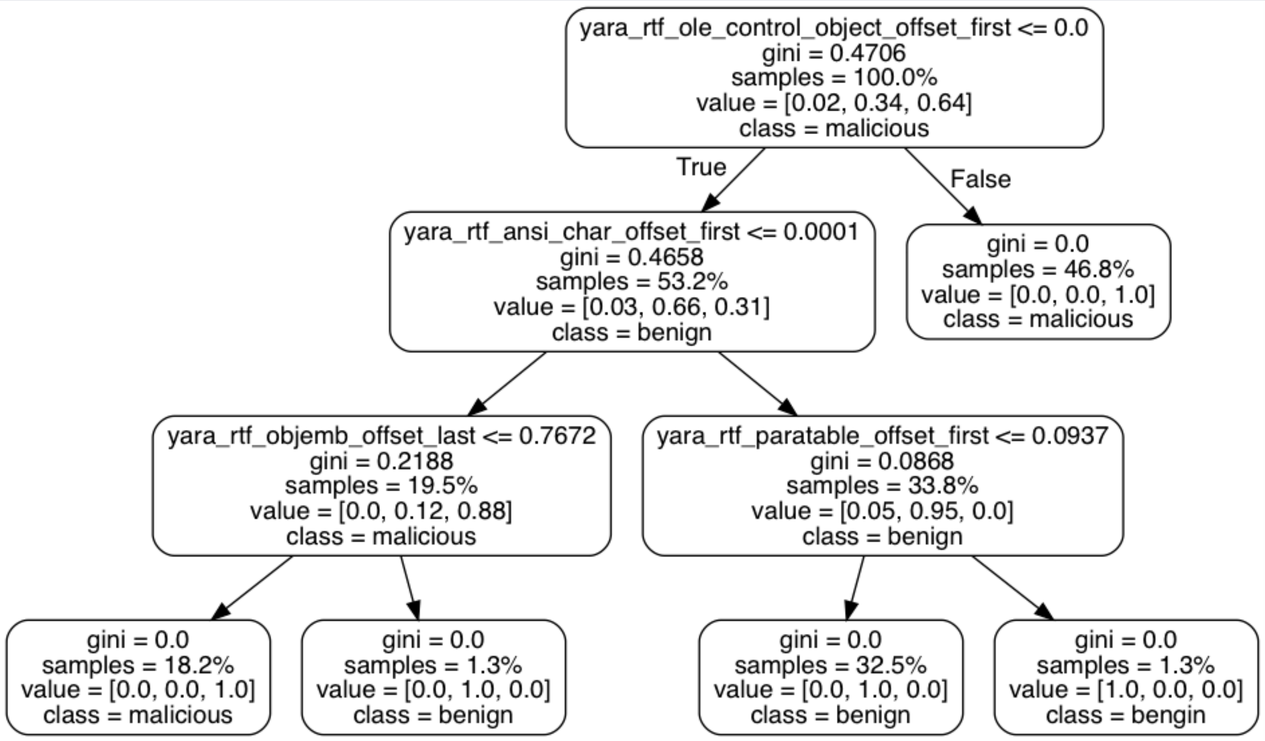
This is hard!

- We need some “ground truth” to train our algorithm
- Trust a system or do analysis?
- What is malicious?



Random Forests

- Small labeled dataset, mimic anal



rule	weight
yara_rtf_ole_control_object_offset	0.084104
yara_rtf_ole_control_object	0.067022
yara_rtf_ansi_char_offset	0.054197
yara_rtf_objdata_offset	0.044908
yara_rtf_empty_word_offset	0.043165
yara_rtf_high_binary_run	0.037604
yara_rtf_empty_word	0.037285
yara_rtf_objdata	0.036719
yara_rtf_ansi_code_page_char_offset	0.029801
yara_rtf_ansi_char	0.027635
yara_rtf_invalid_control_tag_offset	0.024327
yara_rtf_colortable_offset	0.022953
yara_rtf_invalid_control_tag	0.022599
yara_rtf_viewkind_offset	0.020068
yara_rtf_high_binary_run_offset	0.017897
yara_rtf_ansi_code_page_char	0.016537
yara_rtf_sv_offset	0.014489
yara_rtf_listviewctl_object_offset	0.014378
yara_rtf_word_class	0.013871
yara_rtf_embedded_ole_offset	0.012669

Tsar Team Microsoft Office Zero Day CVE-2015-2424

Identification | Details | Content | Analyses | Submissions | ITW | Comments

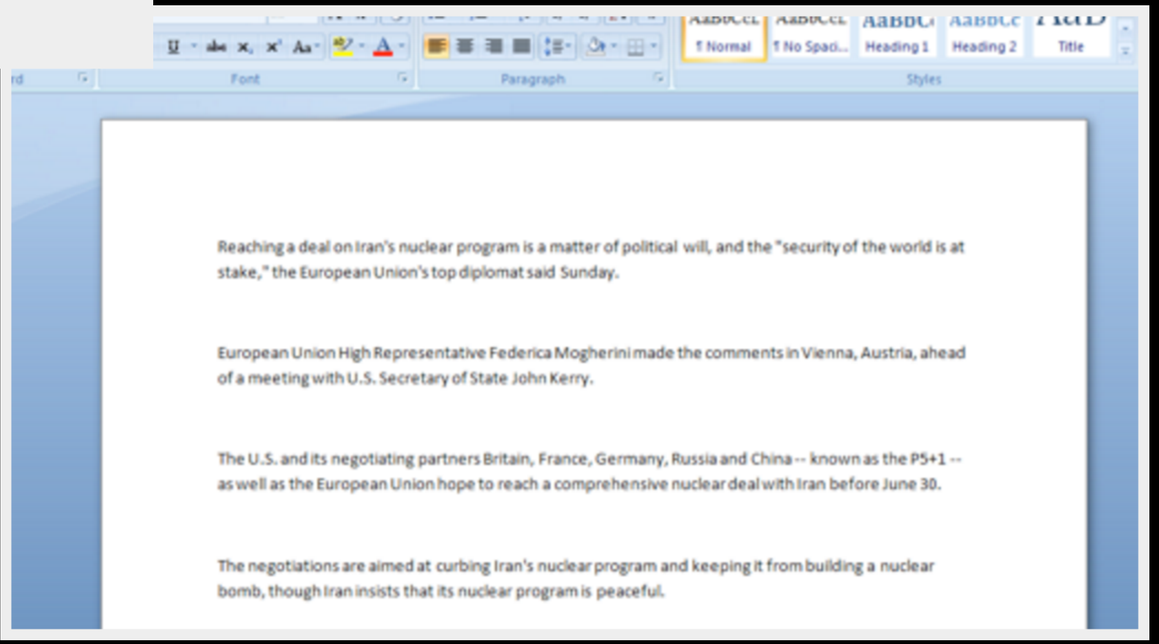
SUPERAntiSpyware	-	5.6.0.1032	20150630
Symantec	-	20141.2.0.56	20150630
Tencent	-	1.0.0.1	20150630
TheHacker	-	6.8.0.5.584	20150630
TrendMicro	HEUR_RTFEXP.A	9.740.0.1012	20150630

2015-07-15 18:58:54 **1/56**

2015-06-30 10:44:29 **1/55**

<code></code>

<code></code>



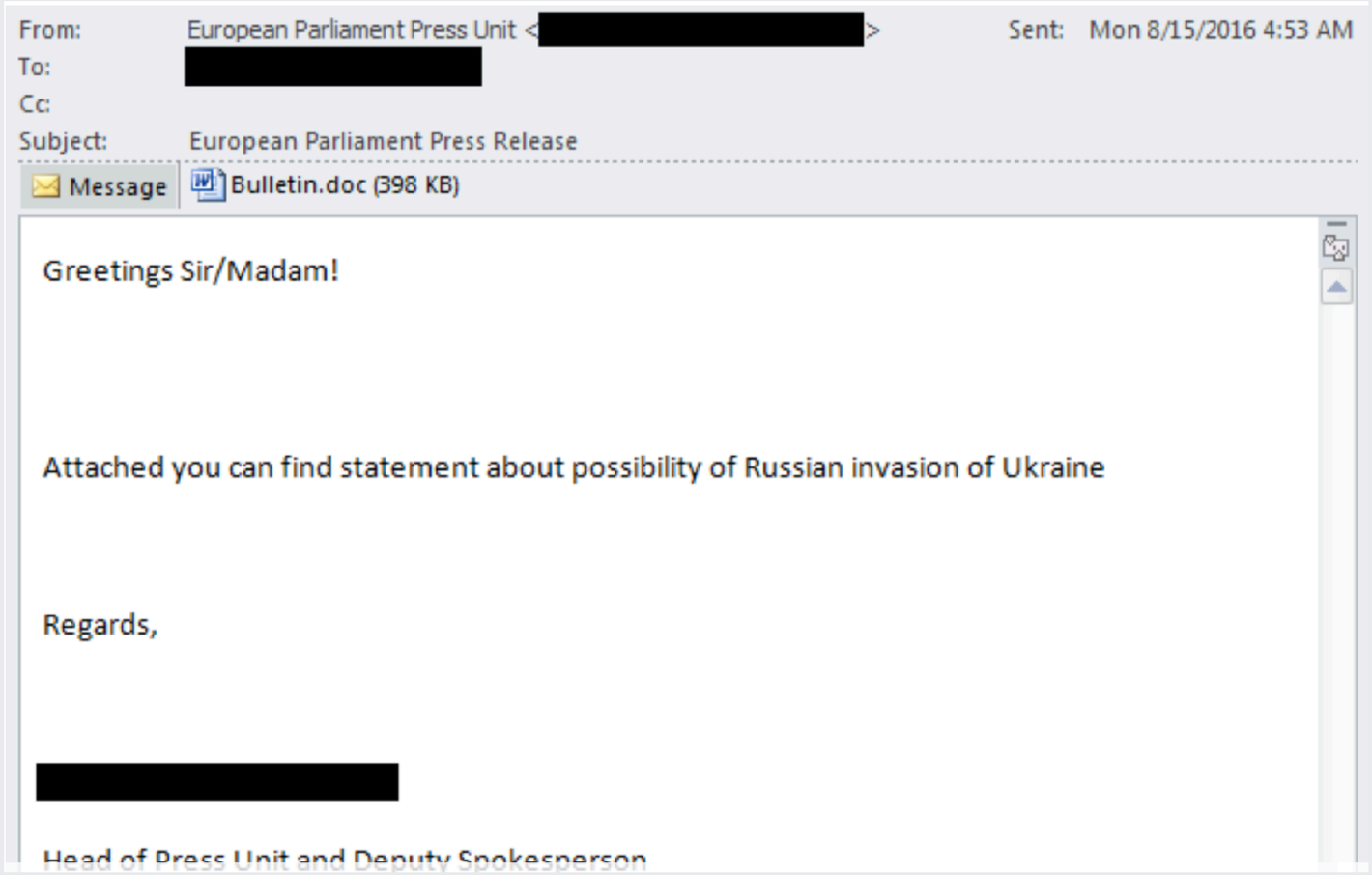

```
md5 = "112c64f7c07a959a1cbff6621850a4ad"
y = YaraScan()
sample_features = pd.DataFrame([get_initial_features(path+sample_prefix+md5, y)])
probs = forest.predict_proba(sample_features[features])
vt_pos = get_vt_pos(md5)
if not vt_pos: vt_pos = 0
print "MD5: %s\nProb Mal: %02f\nProb Benign: %02f\nVT Pos: %d" % (md5, probs[0][1], probs[0][0], vt_pos)
```

```
MD5: 112c64f7c07a959a1cbff6621850a4ad
Prob Mal: 0.814000
Prob Benign: 0.186000
VT Pos: 33
```


MARIO
000000
×00
WORLD
1-1
TIME
SUPER COOL
STORY BRO.
©1985 NINTENDO
1 PLAYER GAME
2 PLAYER GAME
TOP- 000000

‘DealersChoice’ is Sofacy’s Flash Player Exploit Platform

By Robert Falcone and Bryan Lee
October 17, 2016 at 11:00 PM
Category: Unit 42 Tags: adobe, DealersChoice, exploit, Flash Player, Sofacy



From: European Parliament Press Unit <[redacted]>
To: [redacted]
Cc:
Subject: European Parliament Press Release
Sent: Mon 8/15/2016 4:53 AM

Message | Bulletin.doc (398 KB)

Greetings Sir/Madam!

Attached you can find statement about possibility of Russian invasion of Ukraine

Regards,

[redacted]

Head of Press Unit and Deputy Spokesperson

So there I was

thinking - how do I pivot on stuff in RTF's

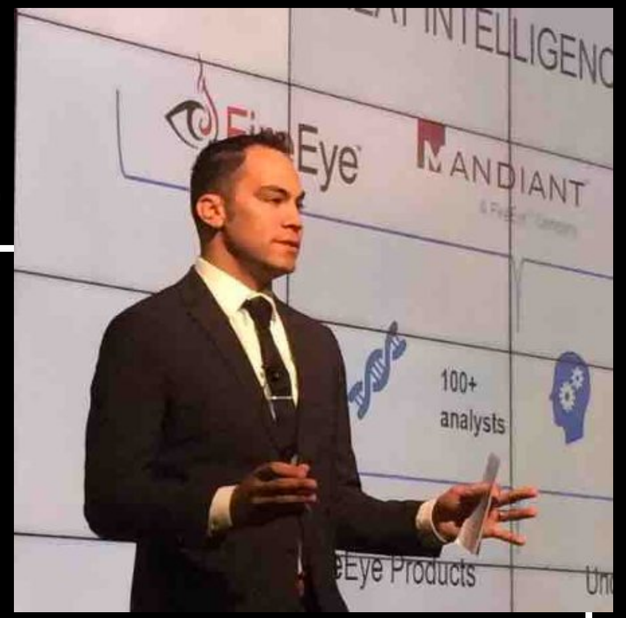
1. Write a RTF parser.
2. Hash and collect lots of things.
3. Analyze groupings of documents for common things.
4. Remove jaw from the floor.

How do RTF Authoring Tools Work?

- Let’s focus on Windows (all the sploits)
- Wordpad = RTF 1.5
  - limited capabilities, no OLE
- Office Word - RTF 1.9.1
  - full drag and drop support OLE

RTF Objects

How is an non-renderable object displayed?




```
<object>
{\result {\rtlch \insrsid2962746
{\pict{\*\picprop\
{\sp{\sn fReallyHidden}{\sv 0}}
{\sp{\sn fFakeMaster}{\sv 0}}
{\sp{\sn fCameFromImgDummy}{\sv 0}}
\wmetafile8\bliptag443765970\blipupi96
{\*\blipuid 1a7354d2647ee40ec69876e0af6edc4a}
```

What are these artifacts?

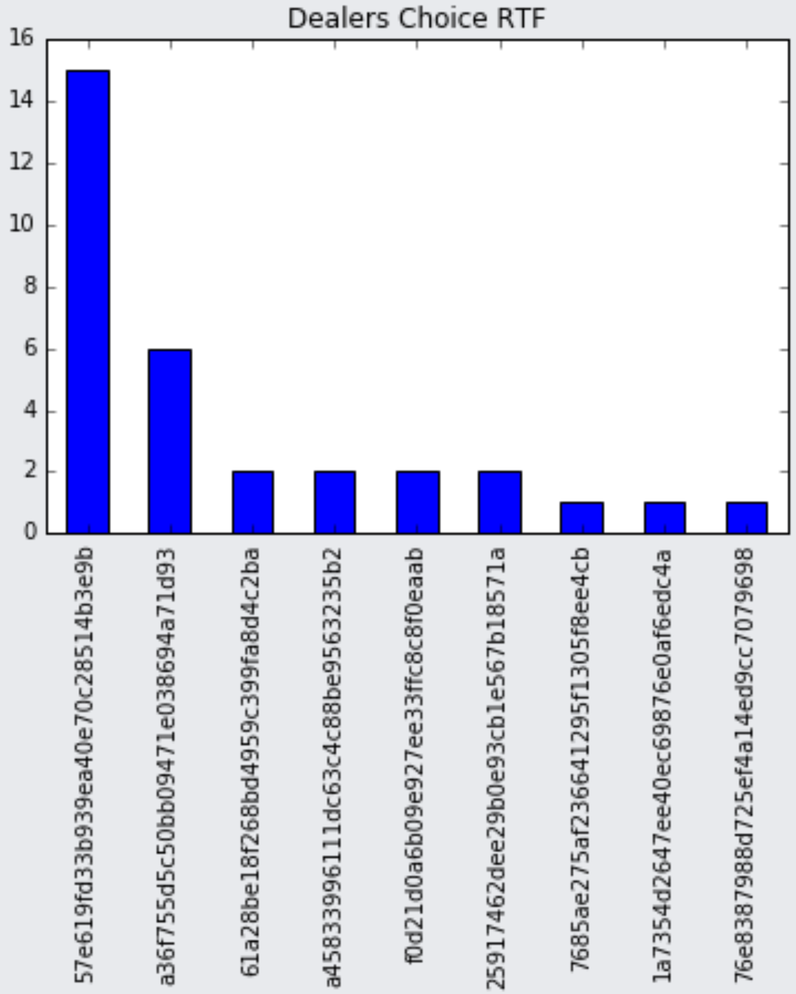
Slide Subtitle

- blipuid - **machine specific** 16-byte ID value (Office)
  - no prescribed method for generation
- insrsid - **session specific** 32-bit ID (Office)

\insrsid*N* 2002	Track Changes (Revision Marks)	Value
\blipuid	Destination of the form '{*' **\blipuid** *XXXX* '}' where *XXXX* is a 16-byte identification number for the image.	

How do you know this isn’t just a weaponizer?

- When you insert an object 2 things are recorded
  -
  - RSID of the session that inserted
  - BLIPUID of the machine that inserted
- Look at the BLIPUID/RSID pair
  - same - likely direct copy
  - different RSID same BLIPUID - different



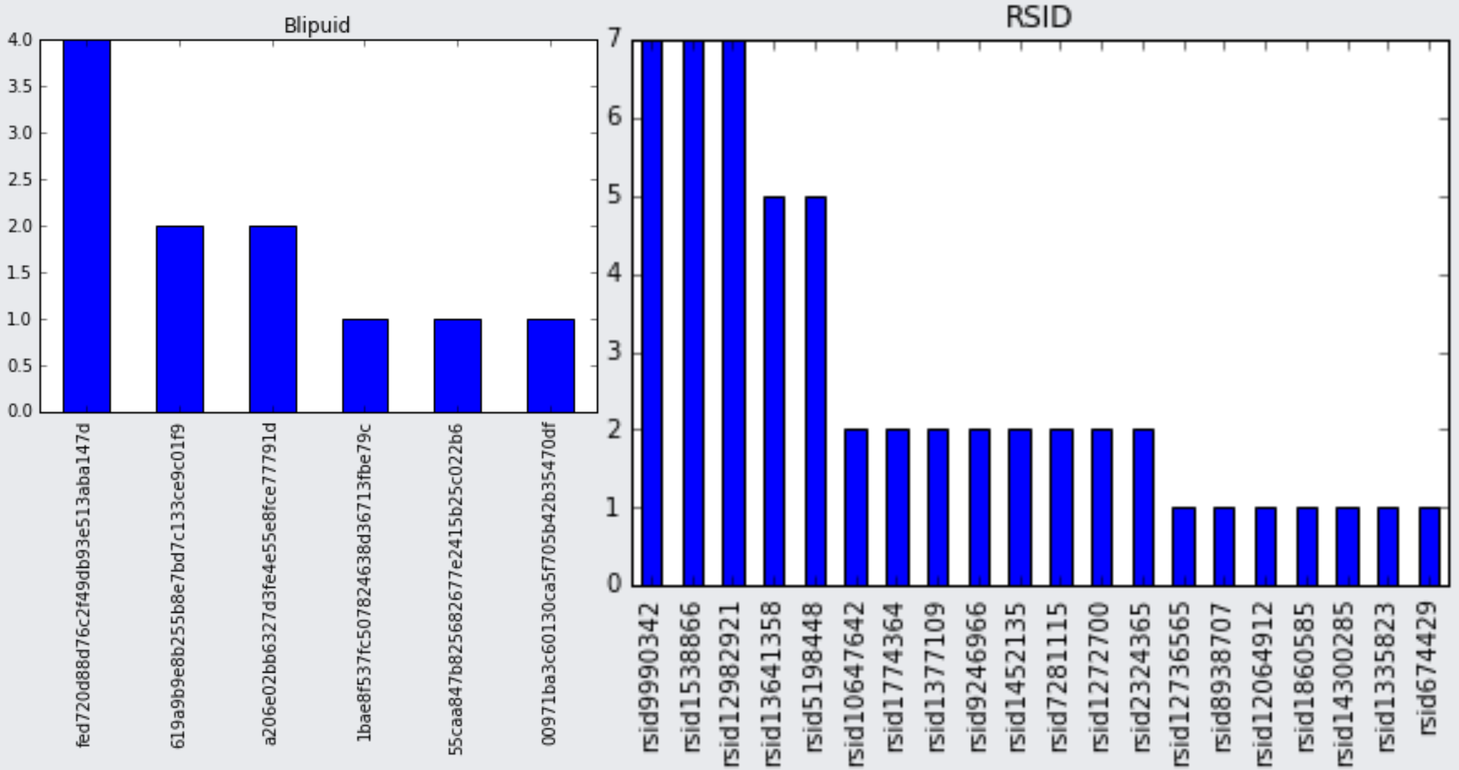
Doc	Date	RSID	BlipUID
Bulletin.doc	8/15/2016	insrsid3488739	a66ce72bb3c6f025aa38614f581dcf1f
word.doc	9/26/2016	insrsid15083000	57e619fd33b939ea40e70c28514b3e9b
Operation_in_Mosul.rtf	10/31/2016	insrsid5270825	8ca97a10b211c57360ad3a2a9c13bc50
NASAMS.doc	11/1/2016	insrsid13654808	57e619fd33b939ea40e70c28514b3e9b
???? (small changes from NASSAMS.doc)	11/2/2016	insrsid13654808	57e619fd33b939ea40e70c28514b3e9b
????	11/8/2016	insrsid1050871	57e619fd33b939ea40e70c28514b3e9b
DGI2017.doc	11/11/2016	insrsid15222711	57e619fd33b939ea40e70c28514b3e9b
Olympic-Agenda-2020-20-20-Recommendations.doc	12/1/2016	insrsid9588558	57e619fd33b939ea40e70c28514b3e9b
OC_PSO_2017.doc	12/2/2016	insrsid9714394	57e619fd33b939ea40e70c28514b3e9b
ARM-NATO_ENGLISH_30_NOV_2016.doc	12/5/2016	insrsid5206680	57e619fd33b939ea40e70c28514b3e9b
Programm_Details.doc	12/15/2016	insrsid9332957	57e619fd33b939ea40e70c28514b3e9b
Program-Submarine_Conference.doc	12/27/2016	insrsid2517161	57e619fd33b939ea40e70c28514b3e9b
NATO Secretary meeting.doc	12/27/2016	insrsid944553	57e619fd33b939ea40e70c28514b3e9b

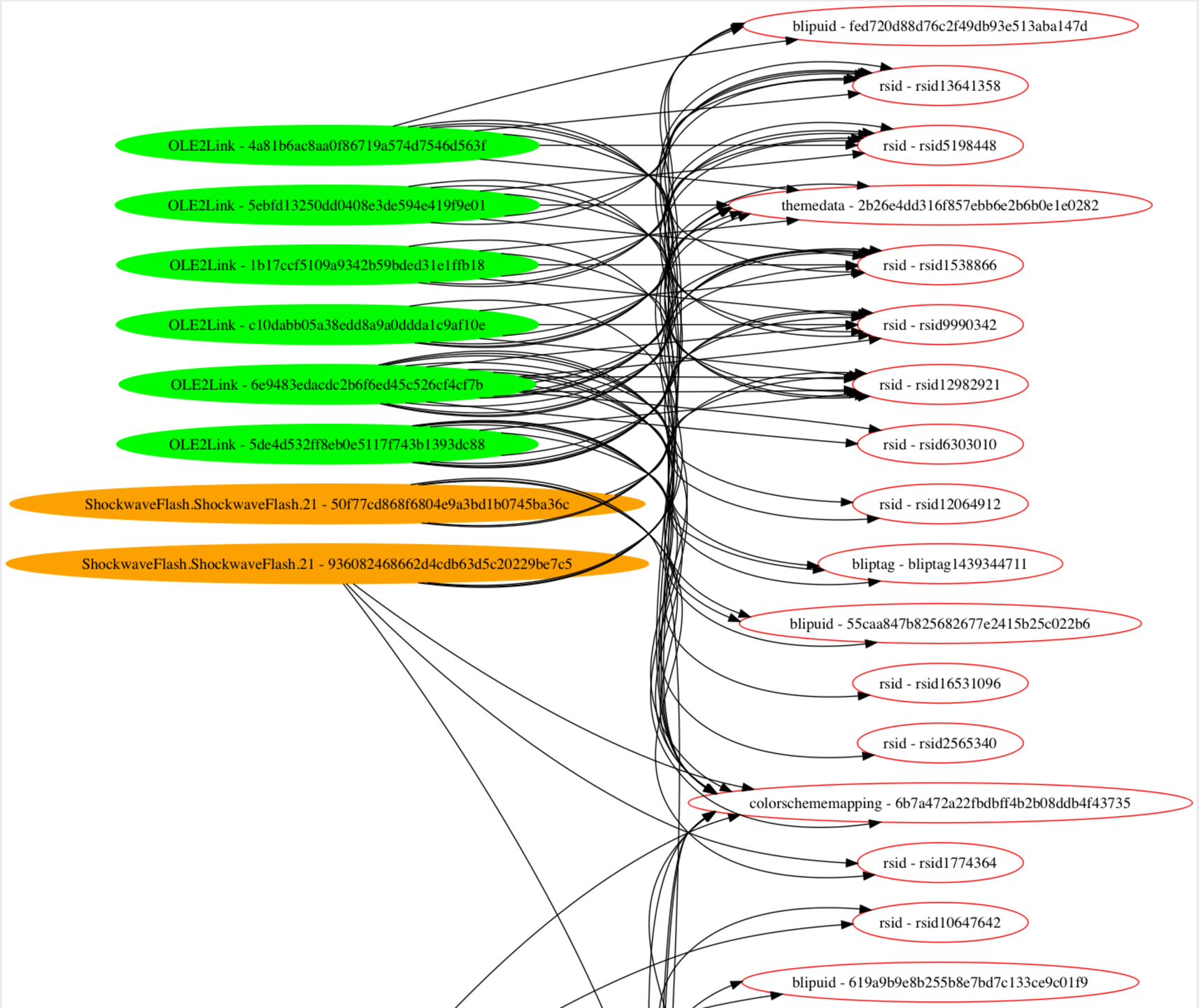
```
rule dealers_choice
{
strings:
   $a = "{\\*\\blipuid 57e619fd33b939ea40e70c28514b3e9b}"
condition:
   $a
}
```

Case #2 - Finfisher / Commercial

May 2016-present

```
Rtf Meta Info:
        create_time: 2013-03-15 01:47:00
        revise_time: 2016-11-27 22:42:00
    internal_version: 66602
           no_chars: 2145
          edit_mins: 1
           operator: tpmfxe
           no_words: 165
           no_pages: 1
         nofcharsws: 1980
             author: tpmfxe
            version: 15
```




blipuid - fed720d88d76c2f49db93e513aba147d
rsid - rsid13641358
OLE2Link - 4a81b6ac8aa0f86719a574d7546d563f
rsid - rsid5198448
OLE2Link - 5ebfd13250dd0408e3de594e419f9e01
themedata - 2b26e4dd316f857ebb6e2b6b0e1e0282
OLE2Link - 1b17ccf5109a9342b59bded31e1ffb18
rsid - rsid1538866
OLE2Link - c10dabb05a38edd8a9a0ddda1c9af10e
rsid - rsid9990342
OLE2Link - 6e9483edacdc2b6f6ed45c526cf4cf7b
rsid - rsid12982921
OLE2Link - 5de4d532ff8eb0e5117f743b1393dc88
rsid - rsid6303010
ShockwaveFlash.ShockwaveFlash.21 - 50f77cd868f6804e9a3bd1b0745ba36c
rsid - rsid12064912
ShockwaveFlash.ShockwaveFlash.21 - 936082468662d4cdb63d5c20229be7c5
bliptag - bliptag1439344711
blipuid - 55caa847b825682677e2415b25c022b6
rsid - rsid16531096
rsid - rsid2565340
colorschememapping - 6b7a472a22fbdbff4b2b08ddb4f43735
rsid - rsid1774364
rsid - rsid10647642
blipuid - 619a9b9e8b255b8e7bd7c133ce9c01f9

Case #2 - commonalities

- consistent metadata
  - random [a-z]{5,10} author
  - edmins1 (from quick edit to dr4gdr0p)
  - old create date (seems random)
  - recent revise time
- small set of “original” rsids, docs contain new rsids over time
-

Case #2 - sig’ing

```
rule rtf_commercial_exploit_kit_weaponizer
{
strings:
  $a = "{\\rt"
  $s1 = /\{\\creatim\\yr201[0-4]/
  $s6 = /\{\\revtim\\yr201[567]/
  $s2 = /\{\\author [a-z]{5,10}\}/
  $s3 = /\{\\operator [a-z]{5,10}\}/
  $s4 = "\\edmins1"
  $s5 = "{\\sn fCameFromImgDummy}"
condition:
  $a at 0 and all of ($s*)
}
```

WTF

Why would bad actors do this? It seems dumb.

- “Normal” mal doc, no result
  - `{\rtf{\object\objocx{\*\objdata`
- Dealers Choice / Hacking Team / CN
  - leverage real documents, always opens
  - drop malicious Flash object in using Word
  - evidence in documents (RSID+BLIPUID)

Applications

- Track lineage of a document
  - benign document -> malicious
- Honeypot signatures
  - create document with specific attributes, look for new RSID
- Indexing all RTF’s in Virustotal for studies
- Automatic signature creation

Next?

Slide Subtitle

- https://github.com/mrichard91/rtflearn
- CRITs + Laika services for rtf_parser