KIZZLE
A SIGNATURE COMPILER FOR DETECTING EXPLOIT KITS

Ben Stock (CISPA, Saarland University)
Ben Livshits (Microsoft Research)
Ben Zorn (Microsoft Research)

/Ben [chiklnorstvz]{4,8}/i
EXPLOIT KITS: CONSOLIDATING MALWARE PRODUCTION

2006

2016

Ben Stock – Kizzle: A Signature Compiler for Detecting Exploit Kits
EXPLOIT KITS (EKS) PROVIDE AN “EXPLOIT-AS-A-SERVICE”

CVE-2012-1876: Recent update to the Cool Exploit Kit landing page

A recently debuted exploit kit (EK), called “Cool EK,” has been known to include various exploits targeting Windows kernel-mode drivers. If you’re unlucky enough to encounter all these exploits in the one place, turned on your computer.

Recently there was an update to the kit’s armaments identified as CVE-2012-0755, CVE-2013-0634, and CVE-2013-2551, which allows for more efficient and effective exploitation.

Neutrino exploit kit now also serves @VUPEN’s CVE-2013-2551 to exploit MSIE 10-9-8-7-6 via the VML integer overflow @kafeine

Targeted Attack

Public Exploit Framework

Criminal Exploit Kit

Internet Explorer Exploit added to infamous exploit pack

An Internet Explorer exploit utilized in high-profile watering hole attacks has been included with the widely used Nuclear exploit pack.

Ben Stock – Kizzle: A Signature Compiler for Detecting Exploit Kits
STRUCTURE OF A TYPICAL EXPLOIT KIT

Unpacker(key)
- PluginDetect
  - CVE-2016-4117
  - CVE-2014-0497
  - CVE-2013-2423

Unpacker(key₂)
- PluginDetect
  - CVE-2016-4117
  - CVE-2014-0497
  - CVE-2013-2423

Unpacker₂(key₃)
- PluginDetect
  - CVE-2016-4117
  - CVE-2014-0497
  - CVE-2013-2423

Unpacker₃(key₄)
- PluginDetect
  - CVE-2016-4117
  - CVE-2014-0497
  - CVE-2013-2423
MALWARE ECOSYSTEM AND AV SIGNATURES

Research

Initial detection

Signature creation

Testing

Signature detection

AV avoiding variant identified

AV avoiding variant deployed

Signature deployment
ADVERSERIAL CYCLE FOR EXPLOIT KITS

Malware Deployment → Initial Detection (days) → Signature Creation (days) → Signature Deployment (days) → Attack Detection (hours) → New Malware Variant (hours) → Malware Deployment (days)

Ben Stock – Kizzle: A Signature Compiler for Detecting Exploit Kits
EXPLOIT KIT EVOLUTION: NUCLEAR

Ben Stock – Kizzle: A Signature Compiler for Detecting Exploit Kits
A GREAT DEAL OF CODE REUSE
EXPLOIT KITS: KEY INSIGHTS

Rate of change differs between the layers
- Rapid changes of the packer
- Very slow changes to the actual payload

New exploits are added, old ones rarely removed

Code is "borrowed" from other kits
Exploit kits change the malware ecosystem in ways that benefit the attackers but also make them easier to detect

- Code in EKs changes relatively slowly and in predictable ways
- A single EK sample can be used to label a related EK samples automatically

Kizzle: automatic detection of EK malware and generation of AV signatures

- Goal: respond to malware mutation automatically in hours instead of days
- Goal: false positive and negative rates comparable to human-authored signature
ABSTRACTION

Capture the normalized syntax of a script
- Ignoring randomized variable names

Reduce JavaScript to its structure
- Tokenization

Edit distance as measurement of similarity
- For long enough streams of tokens

```
var Euur1V = this [
  "19D"
] ("ev#333399a1"演习)
```

Keyword Identifier Punctuator
Identifier Punctuator String Punctuator Punctuator String Punctuator
Ben Stock – Kizzle: A Signature Compiler for Detecting Exploit Kits
Ben Stock – Kizzle: A Signature Compiler for Detecting Exploit Kits
LABELING UNPACKED SAMPLES

- Fast computation of similarity using “Winnowing”
- Used by MOSS, a system to detect plagiarism
LONGEST COMMON TOKEN SEQUENCE

KPPKPPPPPKKIIIKIPIPPPPPIIPKPPKPKPKIIKKPIKPKIPIKIIK

IPPKIIPIKIPKIIKIIPIPIPPPIIPKIIPIKIIKKKKIIKKIIKIKKKIKIKKPKIIKIIIPI

KPPPIPIKPPPKPPIKPIKPIPIPIPIPIPKKKKPKKIIIKIKPPPIPPKIPPPK
SIGNATURE GENERATION

Euur1V = this [ "19D" ] ( "ev#333399al" ) ;
jkb0hA = this [ "uqA" ] ( "ev#ccff00al" ) ;
QB0Xk = this [ "k3LSC" ] ( "ev#33cc00al" ) ;

[A-Za-z0-9]{5,6} = this \[ [A-Za-z0-9]{3,5} \] \(.{11}\);
EXPERIMENTAL SETUP

Data

A month worth of data: August 2014

Four major exploit kits: Nuclear, Sweet Orange, Angler, RIG

Grayware data obtained from IE11 crawler with ActiveX invocations

Kizzle implementation

Scope-based

50 machines used for initial clustering

Jobs take about 90 minutes on average

For evaluation, we verify AV and Kizzle results to obtain ground truth
EXPERIMENTAL SETUP

Clustering

Minimum of **100 tokens** to be considered for clustering

Maximum distance of **10%**

Minimum **50 members** per cluster

Signature Generation

Maximum of **200 tokens in sequence** to avoid too long signatures

Minimum of **200 chars in signature** to avoid false positives
FALSE POSITIVES

Ben Stock – Kizzle: A Signature Compiler for Detecting Exploit Kits
FALSE NEGATIVES

Ben Stock – Kizzle: A Signature Compiler for Detecting Exploit Kits
VULNERABILITY WINDOW ANGLER

False negatives for Angler

6 days

Ben Stock – Kizzle: A Signature Compiler for Detecting Exploit Kits
EVOLUTION OF KIZZLE SIGNATURES

Graph showing the evolution of Kizzle signatures from August 1st to August 31st. The graph includes lines for RIG, Angler, Sweet orange, and Nuclear, with specific signatures such as NEK.sig1, NEK.sig2, RIG.sig1, RIG.sig2, RIG.sig3, RIG.sig4, ANG.sig1, ANG.sig2, ANG.sig3, UluN, and fber443.
COMPARING SIGNATURES FOR NUCLEAR

Generated by Kizzle

```
(?<var0>[0-9a-zA-Z]{3,6})=\[(?<var1>[0-9a-zA-Z]{3,6})\[(?<var2>[0-9a-zA-Z]'\]{5,8})\]
((?<var3>\.{90})
),\k<var1>\[\k<var2>\]
((?<var4>\.{106})\),\k<var1>\[\k<var2>\]
"_r3fwrwg4e3fwrwg4p3fwrwg4l3fwrwg4a3fwrwg4c3fwrwg4e3fwrwg4"\]var(?<var5>[0-9a-zA-Z]{3,7})
```

Hand-crafted by analyst

```
"("b3fwrwg4g3fwrwg4c3fw\x90 + returna["replace"](/3fwrwg4/g,"");var\x90
```
LIMITATIONS AND FUTURE WORK

Small changes mean new signatures per day
- Merging of existing clusters over multiple days to get more generic signatures

Noise in packers could hinder effectiveness
- combine several smaller signatures into one

EKs could change drastically over night
- Labeling on other features, e.g., runtime behavior
CONCLUSION

EK exhibit a great deal of code sharing and overlap.
This is their key weakness which Kizzle takes advantage of.
Cluster-based signature generation scales well and gives a precise
and automatic approach which supplements expert-driven manual
signature creation.
Using a month of data and applying Kizzle to 4 popular EKs, FP
and FN characteristics are comparable or better than AV
signatures.