

「r2+剣Linux」解析の入門

つるぎ



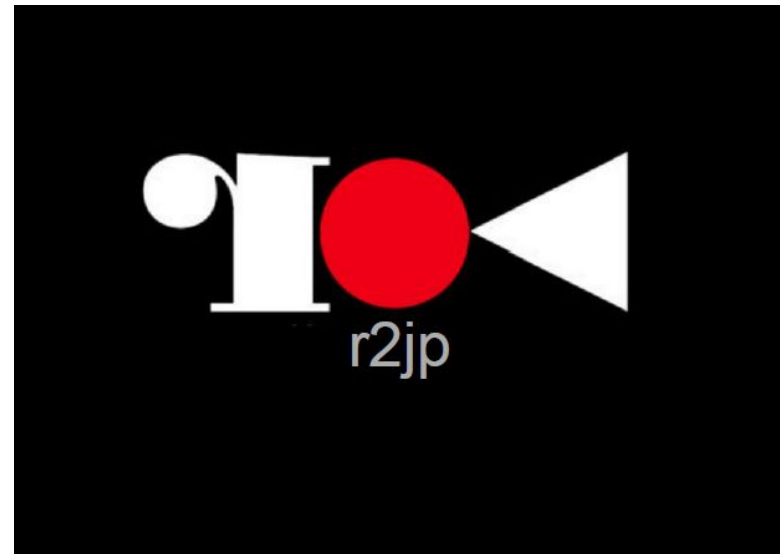
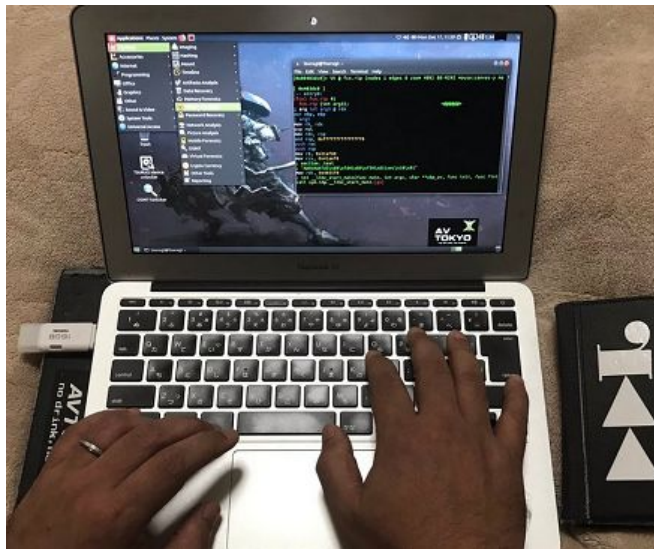
@unixfreaxjp

whoami



- r2jp コミュニティーの人です。
ビルドテストとバグの確認は私の出番です。
 - 2007年からradareという名前からradare2を使い始めて、2011からradareでFreeBSDポートに紹介しました。
 - 現在 r2のdevのテスト環境はx86/amd だけでなくARM/MIPS/SPARC/ARC/PPCのcpuも、さらにOSはFreeBSD/OpenBSD/NetBSD迄。
- 0day.JPとblog.malwaremustdie.orgのマルウェア解析ブログを書く人
- セキュリティキャンプ全国大会の講師 2017～現在
- IPA ICSCoE CISOトレーニングサポーター
- FIRST.orgでのCTI SIGとプログラムコミッティー
- 仕事は(株)ラックのサイバー救急センター

1. 「radare2」 r2でソフトウェア解析入門



@unixfreaxjp



menu

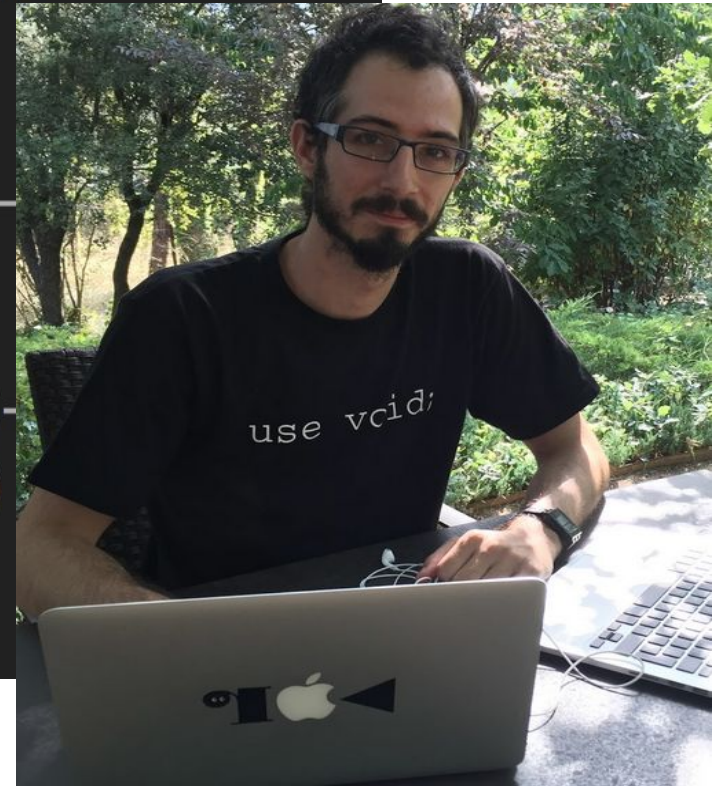
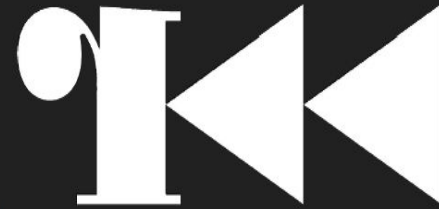
- radare2（読み方：ラ ダ レ）
- リソースの紹介
- r2でソフトウェア解析入門
- R2CON2018



radare2 >> ラダレとは？

Radareとは？

- 12年間開発されているオープンソースプロジェクト(無料)
- リバースエンジニアリングのフレームワークとツールセットで構成
- オリジナル(初期)は私(pancake)が開発
- コミュニティとコントリビューターのコーダーが参加
- 個人による開発スタイルから、プロジェクトリーダー、メンテナー
- 現状は6週間ごとにマイナーバージョンアップをリリース
- +1.0となるメジャーバージョンアップは毎年r2conの後にリリース
- r2conはバルセロナで開催(2017年はおおよそ230人の参加者)
- r2conの全セッションはYouTubeにて公開



radare2 >> OS対応

Architectures

i386, x86-64, ARM, MIPS, PowerPC, SPARC, RISC-V, SH, m68k, AVR, XAP, System Z, XCore, CR16, HPPA, ARC, Blackfin, Z80, H8/300, V810, V850, CRIS, XAP, PIC, LM32, 8051, 6502, i4004, i8080, Propeller, Tricore, Chip8 LH5801, T8200, GameBoy, SNES, MSP430, Xtensa, NIOS II, Dalvik, WebAssembly, MSIL, EBC, TMS320 (c54x, c55x, c55+, c66), Hexagon, Brainfuck, Malbolge, DCPU16

File Formats

ELF, Mach-O, Fatmach-O, PE, PE+, MZ, COFF, OMF, TE, XBE, BIOS/UEFI, Dyldcache, DEX, ART, CGC, Java class, Android boot image, Plan9 executable, ZIMG, MBN/SBL bootloader, ELF coredump, MDMP (Windows minidump), WASM (WebAssembly binary), Commodore VICE emulator, Game Boy (Advance), Nintendo DS ROMs and Nintendo 3DS FIRMs, various filesystems.

Operating Systems

Windows (since XP), GNU/Linux, OS X, [Net|Free|Open]BSD, Android, iOS, OSX, QNX, Solaris, Haiku, FirefoxOS

Radare is a [LGPL](#) portable reversing framework that can:

- Go

Tools

```
Write your Favourite x86-64 opcode....
```


```
0: xor rax,raxret
4310c3




/ (len) entry0 42:
1: entry0:
0x00404040 4310c0 xor rax, rax
0x00404040 23 ret
0x00404040 c310c3 ret
0x00404040 55c2 mov edi, esp
0x00404040 4310c3f0 and rax, 0xffffffffffffff00
0x00404040 59 push rax
0x00404040 54 push rbp
```






[illegible]

©2018, LGPL, pancake. | Page source

radare2 >> "dev板"と"stable版"の違い






 radare / **radare2**

 Unwatch ▾ 446  Unstar 8,975  Fork 1,736


 Code  Issues 1,049  Pull requests 17  Projects 16  Insights

unix-like reverse engineering framework and commandline tools security <http://www.radare.org/>



radare2 c commandline unix reverse-engineering forensics analysis security

 20,292 commits  75 branches  59 releases  588 contributors  LGPL-3.0


Branch: master ▾ [New pull request](#) [Create new file](#) [Upload files](#) [Find file](#) [Clone or download ▾](#)


 kazarmy and radare ie/iee: remove laddr (#12539) Latest commit 249899e Dec 22, 2018


Latest release


 3.1.3
 57dd0b4

3.1.3 - Codename Antiox

 **radare** released this Dec 4, 2018 · 130 [commits](#) to master since this release

 **Assets** 2

 [Source code](#) (zip)

 [Source code](#) (tar.gz)

See <https://github.com/radare/radare2/releases/tag/3.1.2> for more details

radare2 >> 最終リリース

3.1.2
b453df6

3.1.2 - codename Antivox

radare released this Dec 3, 2018 · 142 commits to master since this release

Assets 2

 [Source code](#) (zip)

 [Source code](#) (tar.gz)

Release Notes

Version: 3.1.2

Previous: 3.1.1

Commits: 12

Contributors: 4

Description

This is a bug-fix release, fixing crashes in the x86, arm64 assemblers and the macho parser. But also improving the xrefs visual navigation experience and panels.

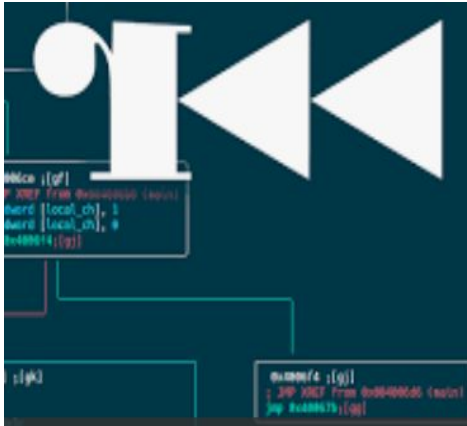
This is the 2nd minor-release after 3.1.0, which also fixed a lot of portability issues. Thanks [@unixfreaxjp](#) for all the testing on freebsd, powerpc, sparc, mips, armv5 spotting out some issues with inline assembly it. Also say thanks to [@revskills](#) and [@HongxuChen](#) for fuzzing and spotting some crashes fixed by [@trufae](#).

Thanks to [@ret2libc](#) for all the refactoring in RBin, which are most of the WIP but setting the base for future directions and code cleanups.

radare2 >> 御協力をお願い！

- Bugレポート ⇒ Issue ⇒ PullReq ⇒ Commit
- 日本語の翻訳
- radare2での解析経験のブログ
- radare2のプラグイン・プロジェクトの開発
- r2jp / radare2 のイベントに参加
- など

RADARE2はオープンソースのソフトウェアです！



リソース >> radareorg/r2jp

r2jp = Japanese Community of radare2

repo/ホームページURL

<https://github.com/radareorg/r2jp>

ロゴ



r2jpは日本国内のradare2コミュニティで、ここでradare2に対しての日本国内ユーザのサポート、イベントと技術交換などのやり取りを行っています。目的は下記の内容をフォローが出来るように強い国内コミュニティを作る前提ですので、ターゲットは例えば下記の対応が出来る迄と考えております。

- 質問/ Q&A (質問があれば新規Issuesのチケットで対応、日本語で書いてください)
- 開発(プラグイン、追加機能、アイデア)
- 日本語ドキュメンテーション
- ワークショップ、トレーニング、プレゼンテーションなど

現在対応方法について

内容によって、radare2ベテランのユーザーが便利な使い方、質問の回答、howto手順などをWikiで書きます。

リソース >> radareorg/r2jp

radare2全般の重要なリンク情報

- site <https://www.radare.org/> or <http://rada.re/>
 - releases <https://github.com/radare/radare2/releases>
 - dev/source: <https://github.com/radare/radare2>
 - doc <http://rada.re/vdoc/>
 - book <https://radare.gitbooks.io/radare2book/content/>
 - documentaton <http://radare.today/posts/radare2-is-documented/>
 - blog <http://radare.today>
 - installers <http://radare.mikellocc.com/list>
 - r2con conference <https://rada.re/con/> or <https://github.com/radareorg/r2con>
 - web demo <http://cloud.rada.re> (今メンテナンス中)
 - ctf tips <http://radare.today/posts/using-radare2/>
 - twitter @radareorg <https://twitter.com/radareorg>
-

r2でソフトウェア解析入門

0. はじめる前に

- 50分で終わらす予定です。御協力をお願いします
- コマンドの説明後にR2コンソールで試してみましょう
- 作成中にコマンドの質問があれば手を上げててください
- 写真はOKですが、オンラインの公開はNG

r2でソフトウェア解析入門

1. インストールのコマンド

Clone repo

```
$ git clone https://github.com/radare/radare2
```

Go to r2 created directory

```
$ cd radare2
```

Install/update

```
$ ./sys/install.sh #automatically pulls last version from git
```

r2でソフトウェア解析入門

2. ロードのコマンド

Open file

```
$ r2 /bin/ls
```

Don't load settings or scripts

```
$ r2 -N /bin/ls
```

Open file in write mode

```
$ r2 -w /bin/ls
```

Alias for r2 malloc://512

```
$ r2 -
```

Open file in debug mode

```
$ r2 -d /bin/ls
```

Open r2 with no file opened

```
$ r2 --
```

r2でソフトウェア解析入門

3. 解析基本のコマンド

- `s` -> seek
- `px` -> print hexdump
- `pd` -> print disassembly
- `wx` -> write hexpairs
- `wa` -> write assembly
- `aa` -> analyse all (code)
- `q` -> quit
- Append `?` to any command to get help about it.
- Temporary seek with `@`.

r2でソフトウェア解析入門

4. 解析追加コマンド

- Append `j` (`j~{}`) for json (indented) output
 - Example: `izj`, `izj~{}`.
- Append `q` for quiet output.
 - Example: `izq`
- Pipe with shell commands.
 - Example: `iz | less`
- Run shell commands with `!` prefix.
 - Example: `!echo hello there r2con2018`
- Internal grep with `~`.
 - `iz~string`

r2でソフトウェア解析入門

5. ビジュアルモードのコマンド (Visual Mode)

- Access visual mode with **V** command:
 - Rotate print mode with **p** command.
 - Press **?** to get visual mode help.
 - Use **:** to run r2 command.
- Access graph view with **VV** command:
 - Really useful to see workflow of functions.
 - Have to be seeked on a function or won't show anything.
 - Move with arrows or **h j k l**.
 - Zoom in/out with **+/-**.

r2でソフトウェア解析入門

6%”%%%. 解析編集環境設定のコマンド (ENVIRONMENT)

Use **e** commands to tune radare2

Add ASM description

```
e asm.describe=true
```

Use UTF-8 chars

```
e scr.utf8=true
```

Enable truecolor

```
e scr.color=3
```

Enable temporary write

```
e io.cache=true
```

You can add **e** commands to ~/.radare2rc file for them to be loaded by default (remember -N to prevent r2 from parsing it).

r2でソフトウェア解析入門

7. デバ깅のコマンド (DEBUG)

- Debugging options under **d** command (Hint: use **d?**):
 - **db** -> set breakpoint
 - **dc** -> continue execution
 - **ds** -> step
- Starts debugging at dynamic loader (not entrypoint).
- Low level debugger. Not aiming to replace source one.
- Tiled visual mode **V!** is extremely useful here.
- Many backends:
 - gdb (in core).
 - r2frida (via r2pm): mem access, bin instrumentation, hooking...

r2でソフトウェア解析入門

8. エミュレーションのコマンド (ESIL)

- Stands for **E**valuable **S**trings **I**ntermediate **L**anguage.
- Standard intermediate language in r2.
- Each instruction is translated into a single string
 - `mov eax 13 -> 33,eax,=`
- Used for emulation and assisted debugging.
- Search expressions, predict jumps, find references.
- **ae** subcommands used to manipulate the VM of ESIL (Hint: use **ae?**).

r2でソフトウェア解析入門

9. もっと細かいコマンド

? → help

i → information

p, s, b,f, @, \$, ?, !, > → print, seek, block, flags, temp, variable, expression, system and redirection

a → Basic Analysis

e → eval variables

C/P → metadata/Projects

/, @@ → search/iterator

w, r, u → write, resize and undo

r2でソフトウェア解析入門

困ったときに...

- <https://www.radare.org/r/docs.html>
- <https://radare.gitbooks.io/radare2book/content>
- `man radare2, rabin2, etc.`
- `? -> Alphabet of commands`
- `V? for visual mode help`

バイナリ解析の練習

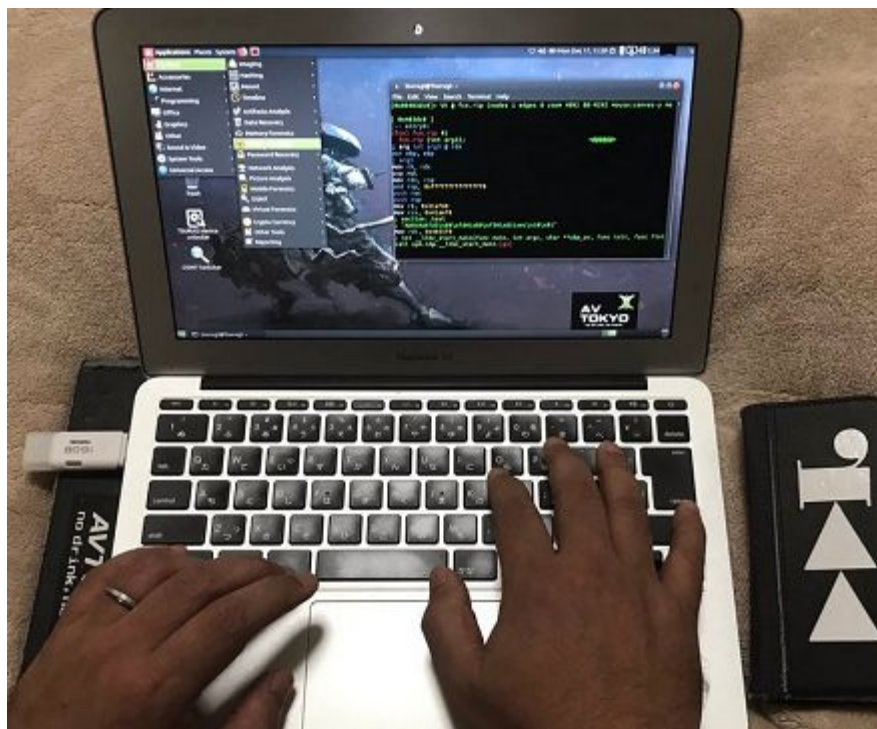
ハッシュ: 1cf8f7df6be98a2de5e956c239c6b8f6

URL: **<http://bit.ly/2CuYWii>**

パスワード: ztrack

ヒント: R2CON2018 / unixfreaxjpのスライド

2. 「Tsurugi Linux」でCTF入門



@unixfreaxjp



Tsurugi(剣)Linux



TS **SURUGI** **Linux**

the sharpest weapon in your DFIR arsenal

最高に切れ味の良い DFIR ツール



Giovanni 'sug4r' Rattaro
@tsurugi_linux

TLP WHITE

tsurugi-linux.org



Tsurugi(剣)Linux

GIOVANNI 'sug4r' RATTARO

- IT SECURITY CONSULTANT @
ITセキュリティコンサルタント
- Italian staff member old <<back|track Linux project
back|track Linux projectのイタリア人古参スタッフ
- Ex developer DEFT Linux
DEFT Linuxの元開発者



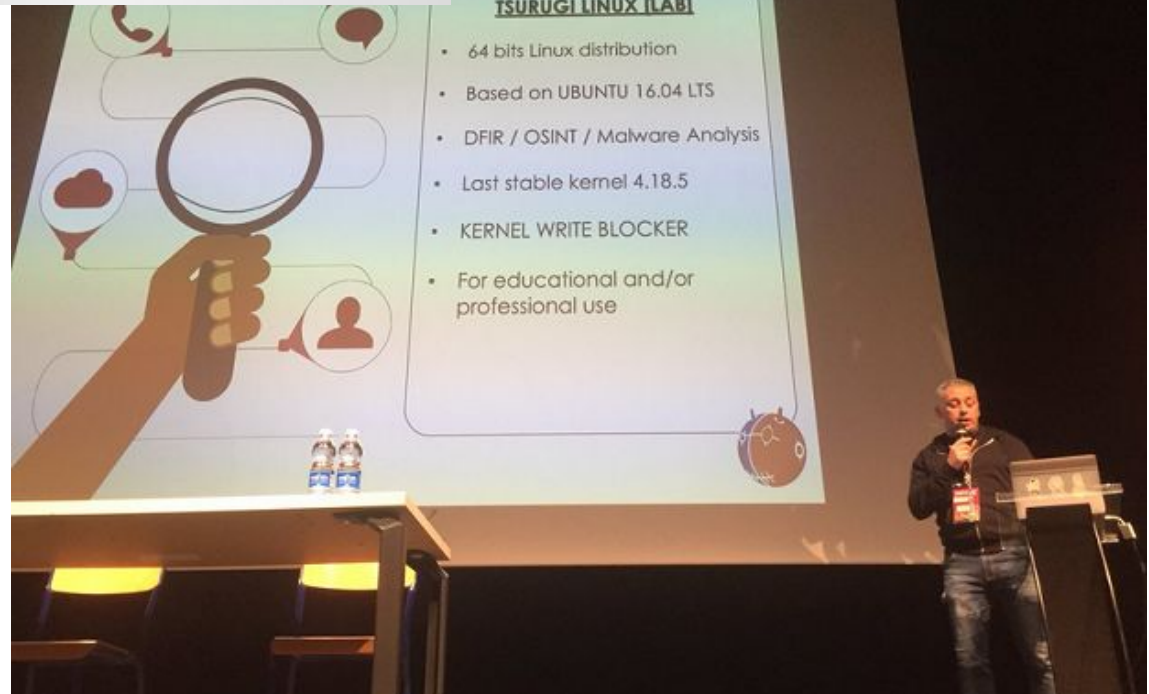
TSURUGI LINUXの開発
チームリーダーは
GIOVANNIさん

(この人↓です)

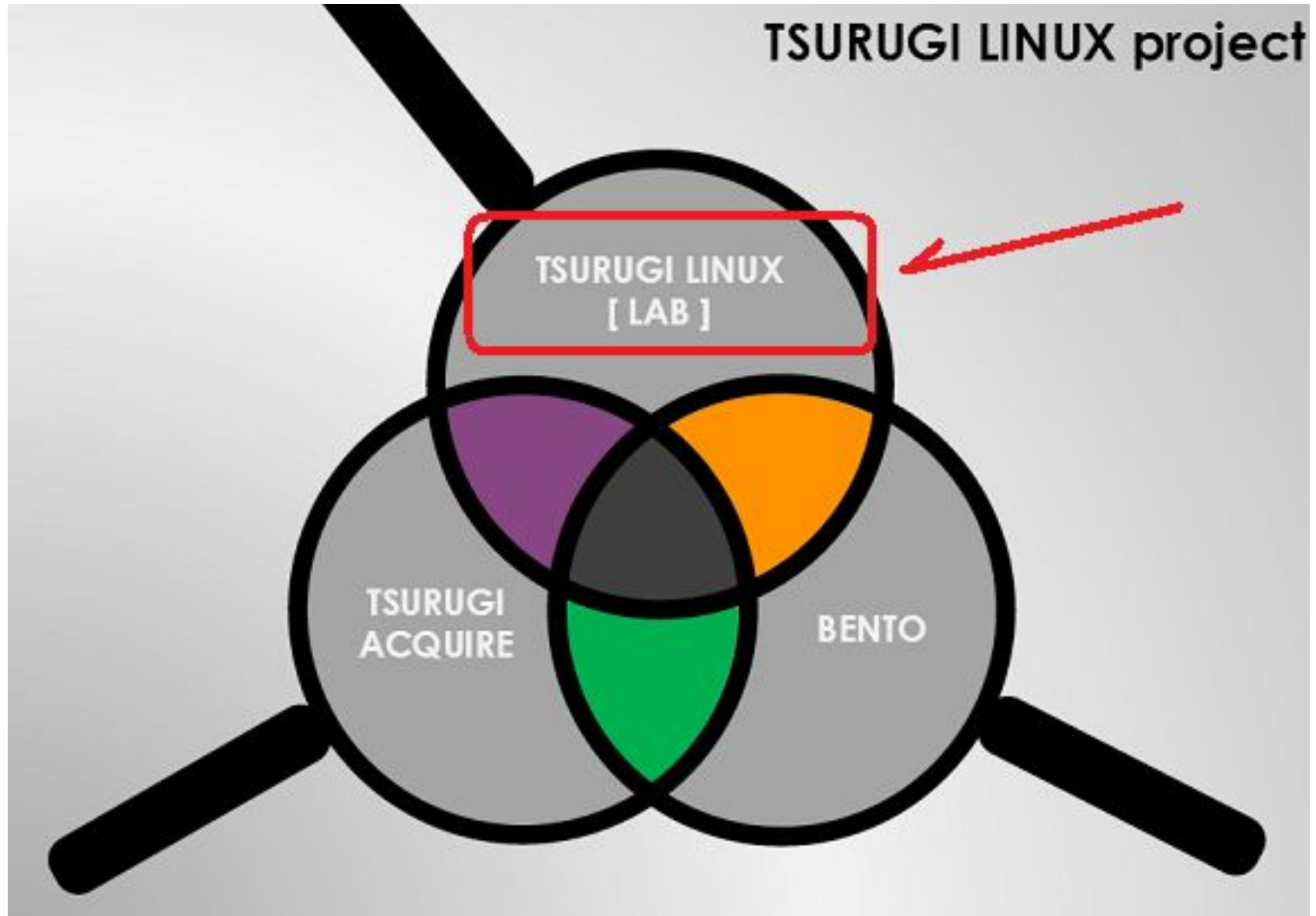
(1) AVTOKYO 2018で
リリースされました。

(2) 現状はLinuxフォ
レンジックツールで一番
人気です

(3) 海外と日本国内の
フォレンジック専門が
作ったツールです。



Tsurugi(剣)Linux



Tsurugi(剣)Linux

HOME

DOWNLOADS

DOCUMENTATION

ABOUT US



TSURUGI

Linux

URL: Tsurugi-Linux.org

The new DFIR Linux distribution



Tsurugi(剣)Linux

TSURUGI LINUX [LAB]

- For educational and/or professional use
教育用/業務用
- DFIR / OSINT / Malware Analysis
DFIR / OSINT / マルウェア 分析
- 64 bits Linux distribution
64ビットLinuxディストリビューション
- Based on UBUNTU 16.04 LTS
UBUNTU16.04 LTSベース
- Last stable kernel 4.18.5
安定版kernel 4.18.5



Tsurugi(剣)Linux

  <https://tsurugi-linux.org/downloads.php>

- DOWNLOAD MIRROR 1 (Temporary unavailable)
- DOWNLOAD MIRROR 2
- DOWNLOAD MIRROR 3
- DOWNLOAD MIRROR 4 (Temporary unavailable)

LAST RELEASES INFORMATION

Tsurugi Linux [64-bit]

Filename: tsurugi_lab_2018.1.iso

sha1: b54895db6fba93239b668edb9f5ef02bef975b40

Tsurugi Acquire [32-bit]

Filename: tsurugi_acquire_2018.1.iso

sha1: f16ad3bc9669efcb22a9859a09598ca12c060d78

Bento portable toolkit

Filename: <soon available>

sha1: <soon available>

Tsurugi(剣)Linux



✝ MalwareMustDie

@malwaremustd1e



ダウンロードが遅いと聞いていますが、剣
LINUX (Tsurugi Linux)のダウンロードが出来
ない方にはワークショップの環境(G-Drive)か
らダウンロードも出来ます↓

bit.ly/2F1moXq

↑本ISOは @AVTOKYO 板となりますので、試
したい方には問題ないですが napdのバグが未
だ残っていて、手で調整が必要



tsurugi_20181011.iso

drive.google.com

79b9b51ac23bd0338a5e67317bca81ceb279907f (SHA1)

使い方

TSURUGI LABの起動選択

1) Liveイメージで起動方法

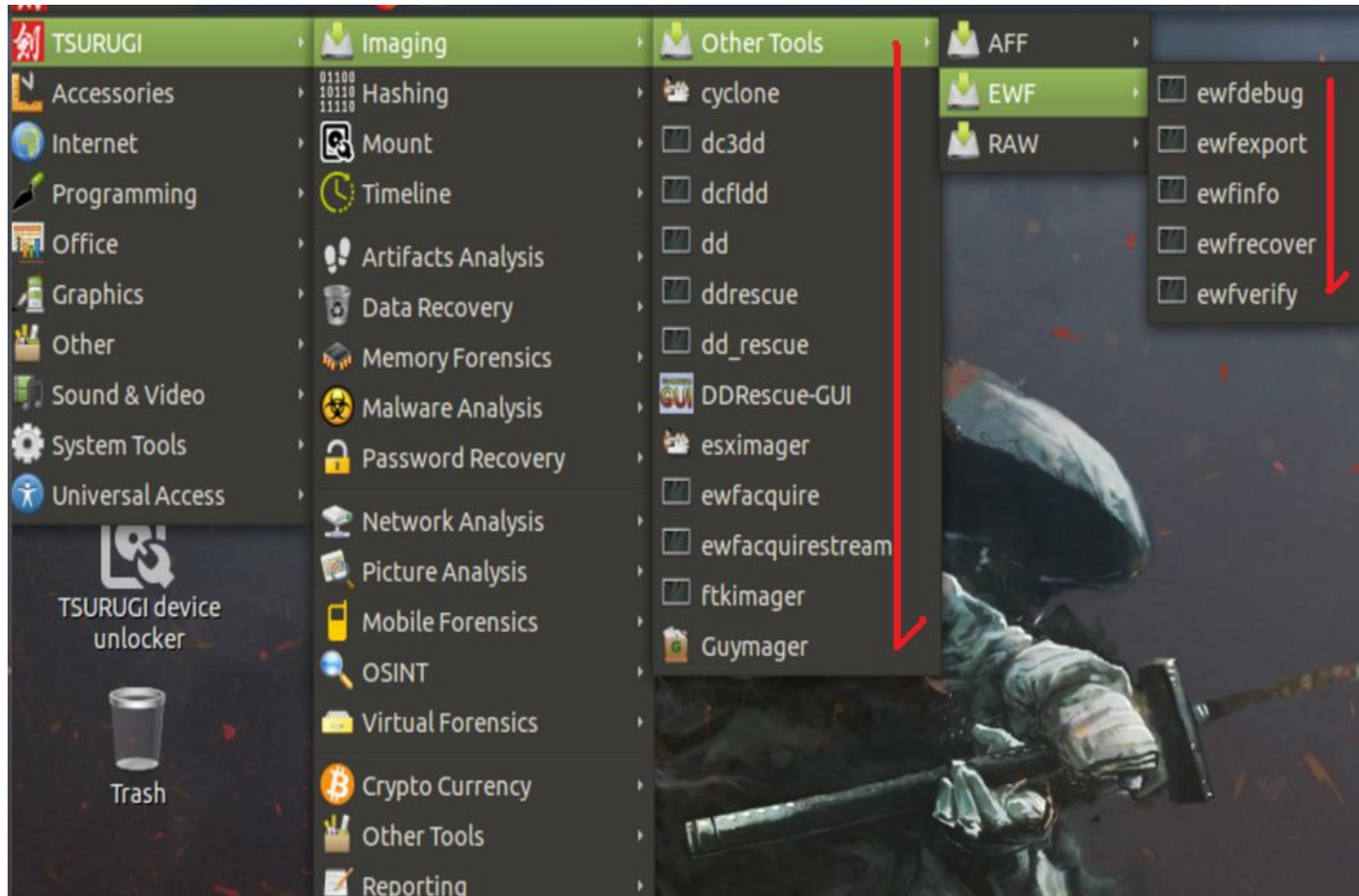
- ・ ISOをダウンロード、ハッシュを確認
- ・ ISOをVmWare又はVirtualBox経由で実行
- ・ ISOをUSBに展開してからUSBブートで使う
- ・ 検体やイメージのアクセスはUSB又はHDDからアクセス
- ・ H/Wおすすめ設定: **RAM/メモリ 4GB と HDD 20GB**

2) インストール手順

- ・ LIVEイメージで起動する
- ・ インストールのアイコンをクリック
- ・ 検体やイメージのアクセスはUSB又はHDDからアクセス

使い方

メニューが沢山あります ↓



合計80件以上のツールがありますので、メニューは良く見ないと欲しいツールが見つからないかも知れない、メニューの形はBACKTRACKのニュアンスです

使い方

TIPS:

- (1) 普通のLinuxの使い方でTsurugiを使ってください。
- (2) 必要がないソフトを動かさない方がいい。
- (3) LIVEを使った時に保存を忘れないでね！
- (4) OSINTの調査はOSINTモードでやりましょう。
- (5) ISOイメージのベースLinuxOSですので、
アップデートがあまりでない。
- (6) LIVE/USBで使う時に新しいH/Wの認識しない。
可能性があります、最新版モデルの対応が弱い。
- (7) たまにサイトと@tsurugi_linuxをチェックしましょう。

フォレンジックCTFの練習

Name ↑	Owner	Last modified by ...	File size
 Challenge1-WinRegistry 	cec 119	—	33 MB
 Challenge2-BrowsingHistory 	cec 119	—	32 KB
 Challenge3-DeletedFile.e01 	cec 119	—	393 MB
 Challenge4-WinMemClipBoard 	cec 119	—	1 GB
 Challenge5-UnknownExecBinary 	cec 119	—	5 KB

URL: <http://bit.ly/2SbMNnH>

フォレンジックCTFの練習

Shared with me > TSURUGI-CTF

Name ↑	Last modified by ...	File size
Challenge1-WinRe...	—	33 MB
Challenge2-Browsi...	—	32 KB
Challenge3-Delete...	—	393 MB
Challenge4-WinMe...	—	1 GB
Challenge5-Unknow...	—	5 KB

- Open with
- New folder...
- Share...
- Get shareable link
- Add to My Drive
- Add to Starred
- Change color
- Search within TSURUGI-CTF
- Rename...
- Download**
- Remove

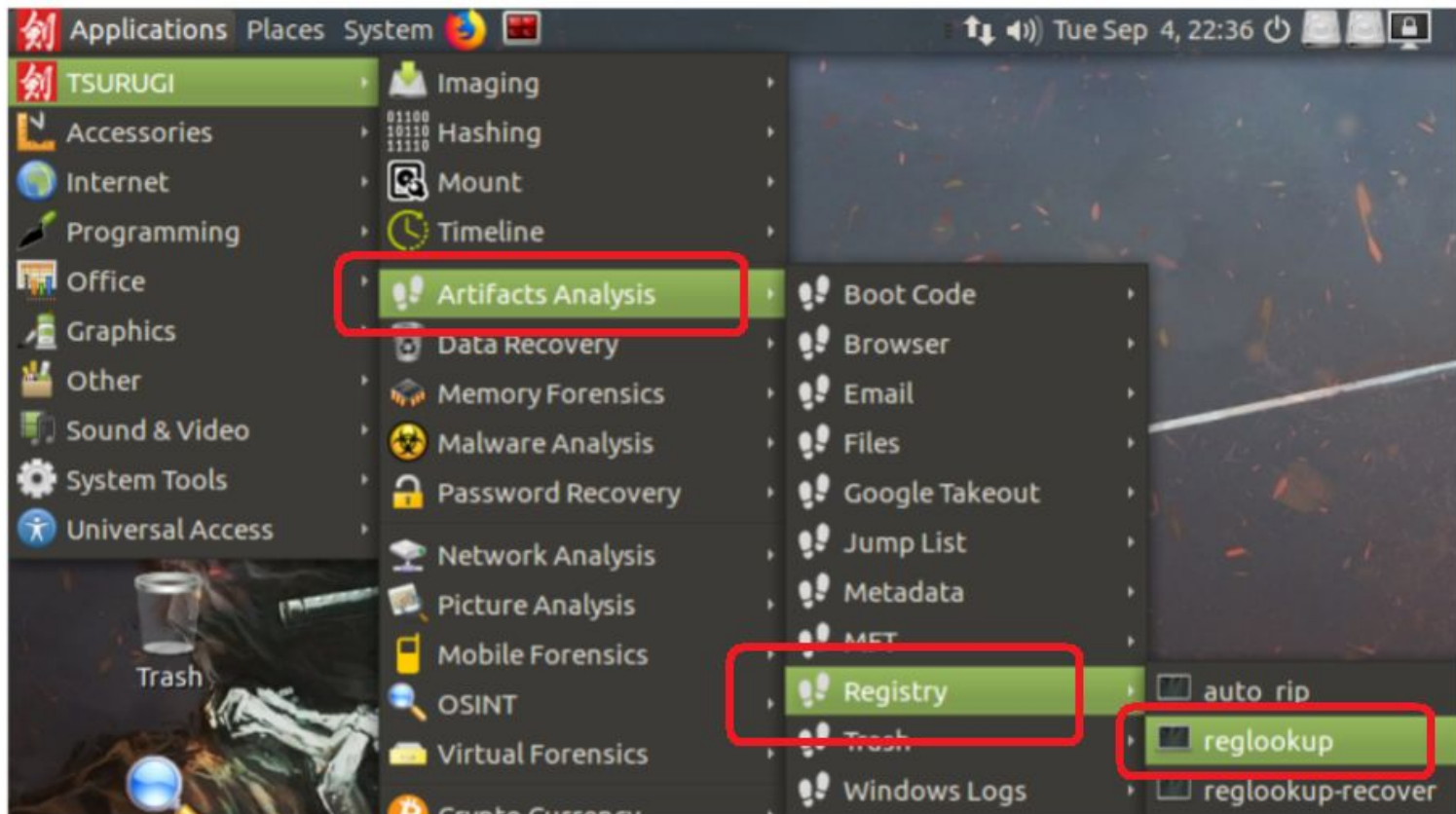
Googleアカウントをログインしたらダウンロードがこんな感じで出来ますよ。

フォレンジックCTFの練習

■レジストリ問題(challenge1)

レジストリデータからフラグを探してください。

使用ツール：reglookup



フォレンジックCTFの練習

```
$ reglookup Challenge1-WinRegistry 2>/dev/null | grep flag
/Classes/CLSID/{7BA4C740-9E81-11CF-99D3-00AA004AE837}/flags,DWORD,0x00000001,
/Classes/CLSID/{C2FBB630-2971-11d1-A18C-00C04FD75D13}/flags,DWORD,0x00000001,
/Classes/CLSID/{C2FBB631-2971-11d1-A18C-00C04FD75D13}/flags,DWORD,0x00000001,
/Classes/CLSID/{D969A300-E7FF-11d0-A93B-00A0C90F2719}/flags,DWORD,0x00000000,
/Microsoft/Windows/AVTOKYO2018/flag,SZ,HackThePlanet,
/Microsoft/Windows NT/CurrentVersion/Print/Printers/Fax/DsSpooler/flags,DWORD,0x00000000,
/Microsoft/Windows NT/CurrentVersion/Print/Printers/Microsoft XPS Document
Writer/DsSpooler/flags,DWORD,0x00000000,
/Microsoft/Windows
NT/CurrentVersion/SoftwareProtectionPlatform/Plugins/Objects/msft:rm%2Falgorithm%2Fflags
%2F1.0,KEY,,2009-07-14 04:41:12
/Microsoft/Windows
NT/CurrentVersion/SoftwareProtectionPlatform/Plugins/Objects/msft:rm%2Falgorithm%2Fflags
%2F1.0/ModuleId,SZ,c42d83ff-5958-4af4-a0dd-ba02fed39662,
/Microsoft/Windows
NT/CurrentVersion/SoftwareProtectionPlatform/Plugins/Objects/msft:rm%2Falgorithm%2Fflags
%2F1.0/IsService,DWORD,0x00000000,
```

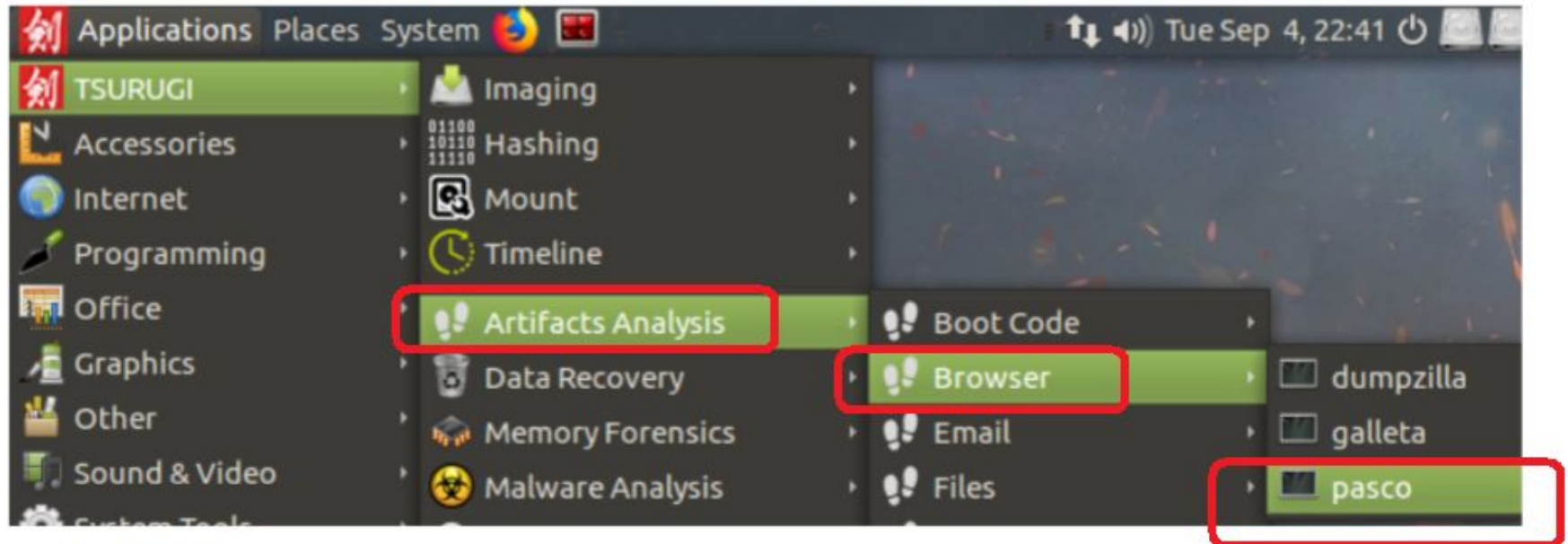
答: HackThePlanet

フォレンジックCTFの練習

■ウェブ閲覧履歴(challenge2)

ウェブ閲覧履歴データからフラグを探してください。

使用ツール：pasco



フォレンジックCTFの練習

\$ pasco Challenge2-BrowsingHistory

History File: index.dat Version: 5.2

TYPE	URL	MODIFIED TIME	ACCESS TIME	FILENAME	DIRECTORY	HTTP HEADERS
------	-----	---------------	-------------	----------	-----------	--------------

URL Visited:	IEUser@	http://www.avtokyo2018.com/flag/is/20181103	09/04/2018 06:23:13			
			09/04/2018 06:23:13			

URL Visited:

IEUser@	http://windows.microsoft.com/en-us/internet-explorer/products/ie-9/welcome	10/23/2013 19:30:06	10/23/2013 19:30:06			
---------	----------------------------------------------------------------------------	---------------------	---------------------	--	--	--

URL Visited:	IEUser@	http://www.avtokyo2018.com/favicon.ico	09/04/2018 06:23:13	09/04/2018 06:23:13		
--------------	---------	----------------------------------------	---------------------	---------------------	--	--

URL Visited:	IEUser@	http://www.avtokyo2018.com	09/04/2018 06:16:14	09/04/2018 06:16:14		
--------------	---------	----------------------------	---------------------	---------------------	--	--

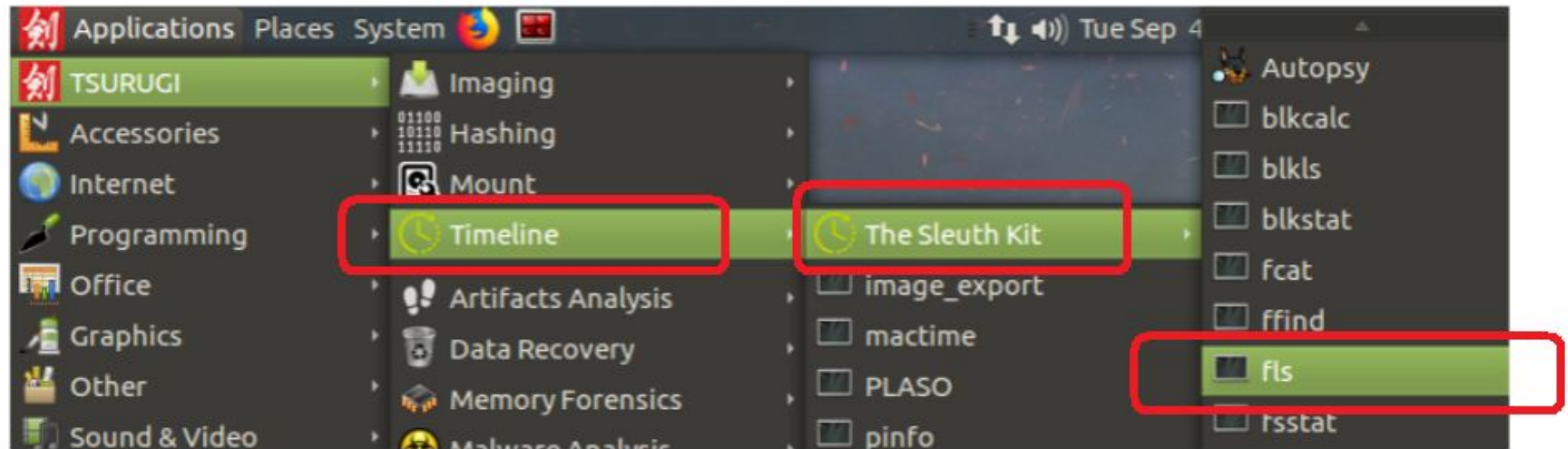
答: 20181103

フォレンジックCTFの練習

■削除ファイル(challenge3)

削除されたファイル名がフラグになっています。

使用ツール：fls



フォレンジックCTFの練習

```
$ fs -d -r Challenge3-DeletedFile.e01 | grep avtokyo  
r/r * 913138:    var/run/console/avtokyo  
r/r * 3880423:  usr/share/man/avtokyo2018/flag/tsurugiDFIR
```

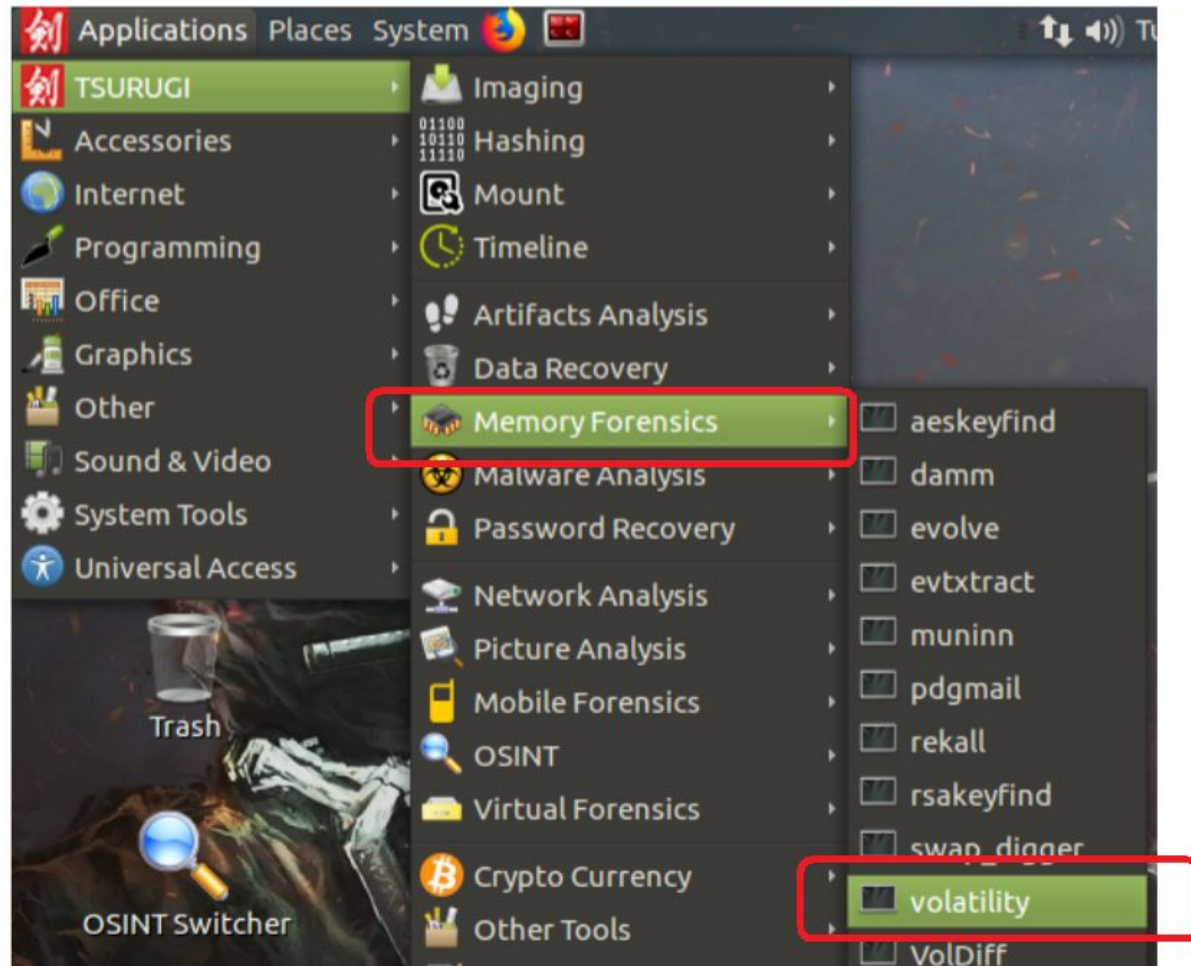
答:tsurugiDFIR

フォレンジックCTFの練習

■メモリ解析/クリップボード(challenge 4)

クリップボードに難読化されたフラグがあります。

使用ツール：volatility□cyberchef



フォレンジックCTFの練習

```
$ volatility -f Challenge4-WinMemClipBoard --profile=Win7SP1x86 clipboard
```

Volatility Foundation Volatility Framework 2.6

Session	WindowStation	Format	Handle	Object	Data
---------	---------------	--------	--------	--------	------

1	WinSta0	CF_UNICODETEXT	0x230173	0xfe5fda68	<u>ZmxhZyBpcvB0c3VydWdp</u>
1	WinSta0	0x0L	0x10	-----	
1	WinSta0	0x3000L	0x0	-----	
1	WinSta0	0x0L	0x3000	-----	
1	-----	-----	0x170067	0xfd1268	

フォレンジックCTFの練習

CyberChefで「ZmxhZyBpcyB0c3VydWdp」をデコード。

The screenshot shows the CyberChef web interface. On the left, the 'Operations' panel lists various tools, with 'From Base64' selected. The 'Recipe' panel shows a single step: 'From Base64'. The 'Alphabet' is set to 'A-Za-z0-9+/' and 'Remove non-alphabet chars' is checked. The 'Input' panel shows the text 'ZmxhZyBpcyB0c3VydWdp' with a length of 20 and 1 line. The 'Output' panel shows the result 'flag is tsurugi' with a time of 18ms, length of 15, and 1 line. A 'Bake!' button is visible at the bottom of the recipe panel.

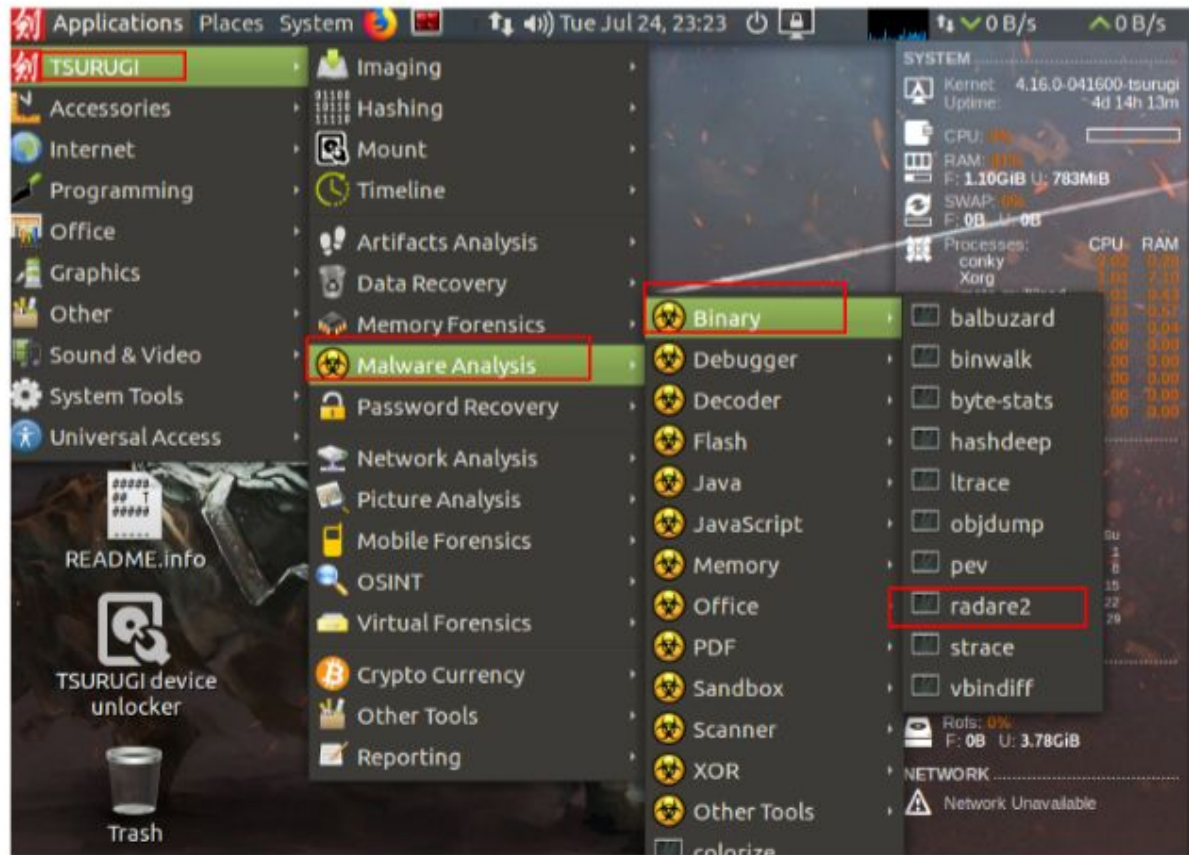
答: flag is **tsurugi**

フォレンジックCTFの練習

バイナリ解析 (challenge5)

正しいFLAGをプログラムに与えてください。

使用ツール：radare2



フォレンジックCTFの練習

下記コマンドでバイナリを読み込みます。

```
$ r2 Challenge5-UnknownExecBinary
```

コマンド実行時の表示

```
— It's the year of radare2 on the desktop
[0x08048360]>
[0x08048360]> af ; pdf
;— entry0:
;— section..text:
;— eip:
(fcn) sym._start 33
  sym._start ();
  0x08048360 31ed      xor ebp, ebp                ; [13] -r-x section size 578 named .text
  0x08048362 5e       pop esi
  0x08048363 89e1     mov ecx, esp
  0x08048365 83e4f0   and esp, 0xffffffff
  0x08048368 50       push eax
  0x08048369 54       push esp
  0x0804836a 52       push edx
  0x0804836b 68a0850408 push sym.__libc_csu_fini ; 0x80485a0
  0x08048370 6830850408 push sym.__libc_csu_init ; 0x8048530 ; "UW1\xffVS\xe8U\xfe\xff\xff\x
  0x08048375 51       push ecx
  0x08048376 56       push esi
  0x08048377 68b0840408 push sym.main            ; 0x80484b0
  0x0804837c e8cfffffff call sym.imp.__libc_start_main ; int __libc_start_main(func main, int a
fini, func rtld_fini, void *stack_end)
[0x08048360]>
[0x08048360]>
```

フォレンジックCTFの練習

メイン関数に移動すると、関数名weird(英語訳: 怪しい、おかしい)が確認できます。
関数「sym.weird」の処理結果が成功メッセージ表示条件になります。|

```
[0x08048360]>
[0x08048360]> s sym.main
[0x080484b0]> af
[0x080484b0]> pdf
(fcn) main 127
main (int argc, char **argv, char **envp);
; var int local_8h @ ebp-0x8
; arg int arg_4h @ esp+0x4
; DATA XREF from sym._start (0x08048377)
0x080484b0      8d4c2404      lea ecx, [arg_4h] ; 4
0x080484b4      83e4f0        and esp, 0xffffffff
0x080484b7      ff71fc        push dword [ecx - 4]
0x080484ba      55            push ebp
0x080484bb      89e5          mov ebp, esp
0x080484bd      53            push ebx
0x080484be      51            push ecx
0x080484bf      89cb          mov ebx, ecx
0x080484c1      83ec0c        sub esp, 0xc
0x080484c4      68c0850408    push str...:_TSURUGI_BINARY_CTF_2018_:: ; 0x80485c0 ; "\n
0x080484c9      e862feffff    call sym.imp.puts
0x080484ce      83c410        add esp, 0x10
0x080484d1      83ec0c        sub esp, 0xc
0x080484d4      68e3850408    push str.Crack_me ; 0x80485e3 ; "Crack me! ; ) "
0x080484d9      e852feffff    call sym.imp.puts
0x080484de      83c410        add esp, 0x10
0x080484e1      833b01        cmp dword [ebx], 1 ; [0x1:4]=-1 ; 1
0x080484e4      7e2a          jle 0x8048510
0x080484e6      8b4304        mov eax, dword [ebx + 4] ; [0x4:4]=-1 ; 4
0x080484e9      83c004        add eax, 4
0x080484ec      8b00          mov eax, dword [eax]
0x080484ee      83ec0c        sub esp, 0xc
0x080484f1      50            push eax
0x080484f2      e864ffff      call sym.weird
0x080484f7      83c410        add esp, 0x10
0x080484fa      85c0          test eax, eax
0x080484fc      7412          je 0x8048510
0x080484fe      83ec0c        sub esp, 0xc
0x08048501      68f1850408    push str.Success ; 0x80485f1 ; "Success!\n"
0x08048506      e825feffff    call sym.imp.puts
```


フォレンジックCTFの練習

関数「sym.weird」に移動します。

関数「sym.weird」にフラグがハードコーディングされています。

```
[0x080484b0]>
[0x080484b0]> s sym.weird
[0x0804845b]> af
[0x0804845b]> pdf
(fcn) sym.weird 85
  sym.weird (int arg_8h);
    ; var int local_90h @ ebp-0x90
    ; var int local_8ch @ ebp-0x8c
    ; var int local_88h @ ebp-0x88
    ; arg int arg_8h @ ebp+0x8
    ; CALL XREF from main (0x80484f2)
    0x0804845b      55          push ebp
    0x0804845c      89e5        mov ebp, esp
    0x0804845e      81ec98000000 sub esp, 0x98
    0x08048464      83ec08      sub esp, 8
    0x08048467      ff7508      push dword [arg_8h]
    0x0804846a      8d8578ffffff lea eax, [local_88h]
    0x08048470      50          push eax
    0x08048471      e8aafeffff  call sym.imp.strcpy          ; char *strcpy(char *dest, const char *src)
    0x08048476      83c410      add esp, 0x10
    0x08048479      c78570ffffff mov dword [local_90h], 0x4f545641 ; 'AVT0'
    0x08048483      c78574ffffff mov dword [local_8ch], 0x4f594b ; 'KY0'
    0x0804848d      83ec08      sub esp, 8
    0x08048490      8d8570ffffff lea eax, [local_90h]
    0x08048496      50          push eax
    0x08048497      8d8578ffffff lea eax, [local_88h]
    0x0804849d      50          push eax
    0x0804849e      e86dfeffff  call sym.imp.strcmp          ; int strcmp(const char *s1, const char *s2)
    0x080484a3      83c410      add esp, 0x10
    0x080484a6      85c0        test eax, eax
    0x080484a8      0f94c0      sete al
    0x080484ab      0fb6c0      movzx eax, al
    0x080484ae      c9          leave
    0x080484af      c3          ret
[0x0804845b]>
```

フォレンジックCTFの練習

```
[  
$  
$ ./challenge5  
  
  .::: TSURUGI BINARY CTF 2018 ::.  
Crack me! ;)  
Wrong!  
  
$  
$  
$ ./challenge5 fugahoge  
  
  .::: TSURUGI BINARY CTF 2018 ::.  
Crack me! ;)  
Wrong!  
  
$  
$  
$ ./challenge5 AVTOKYO  
$  
  .::: TSURUGI BINARY CTF 2018 ::.  
Crack me! ;)  
Success!  
  
$  
$
```

鍵は: “AVTOKYO”

ご質問は？