

Firmwares are weird

A year long journey to efficient
extraction

Black Alps Conference,
November 2022

Quentin Kaiser
quentin.kaiser@onekey.com

NOT YOUR USUAL ANALYST WORKSTATION



NOT YOUR USUAL ANALYST WORKSTATION



NOT YOUR USUAL ANALYST WORKSTATION



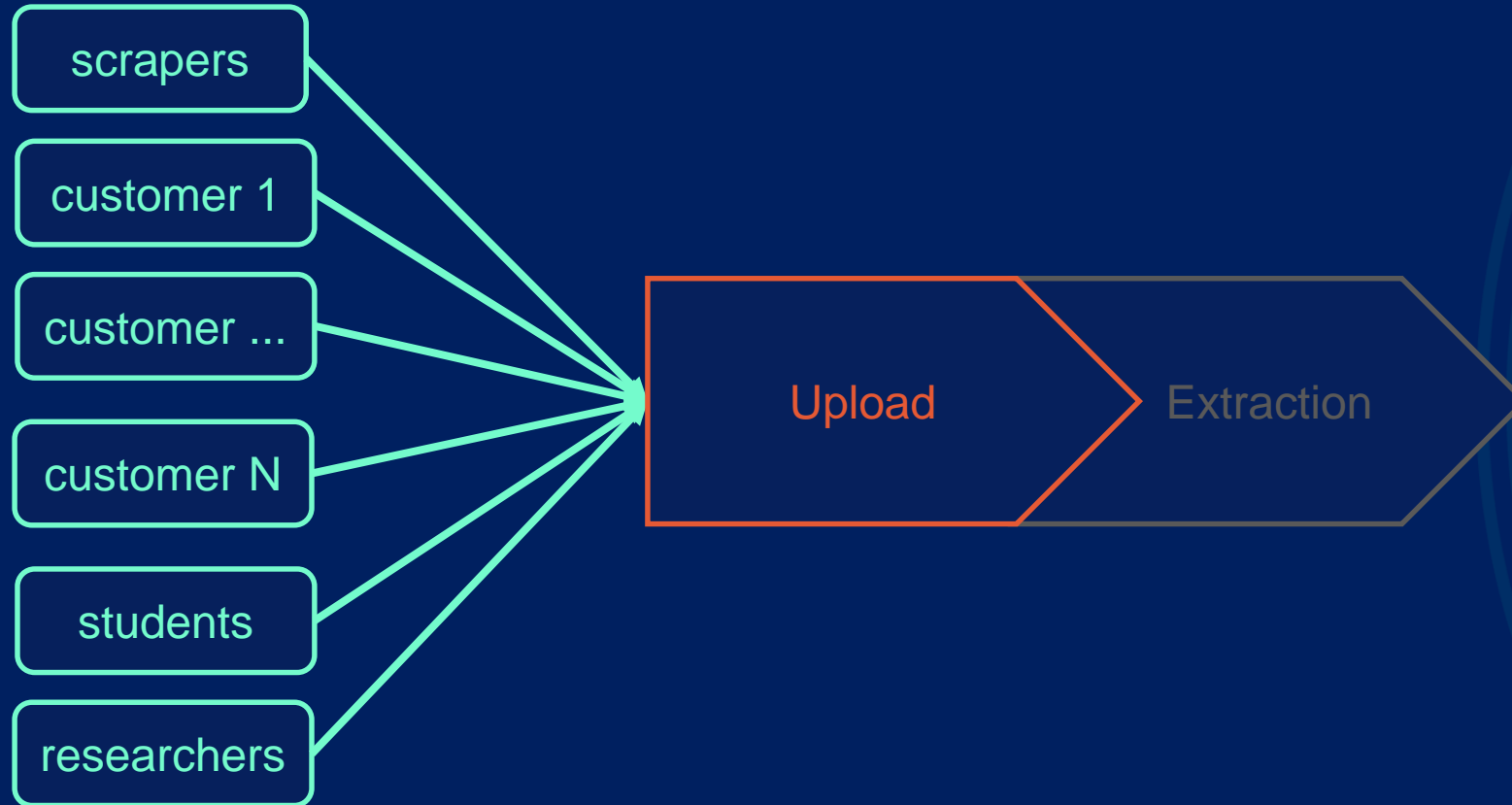
NOT YOUR USUAL ANALYST WORKSTATION



NOT YOUR USUAL ANALYST WORKSTATION

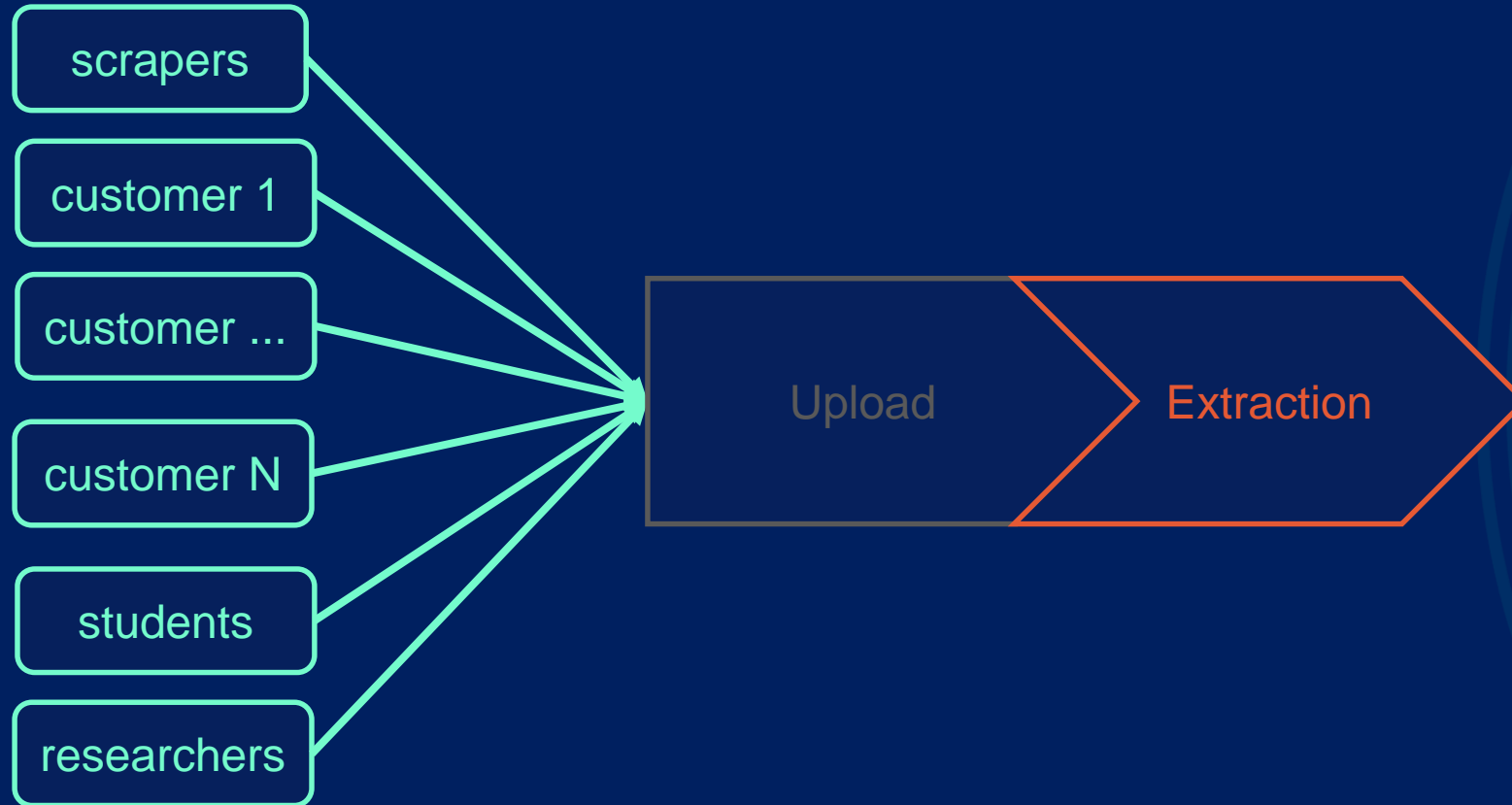


NOT YOUR USUAL ANALYST WORKSTATION



- High amount of uploads
- Concurrency
- Untrusted users
- Source of uploads ?

NOT YOUR USUAL ANALYST WORKSTATION



- Timeouts
- Limited format support
- Memory footprint

We need to automatically extract firmwares of **arbitrary formats**, coming from **untrusted sources**, **at scale**.

Firmware **extraction** framework.

Parses unknown files, matching on more than 30 different **archive**, **compression**, and **file-system** formats.



<https://www.unblob.org>

Clear objectives:

- **Accuracy**
- **Security**
- **Extensibility**
- **Speed**



<https://www.unblob.org>

OBJECTIVE 1 : ACCURACY

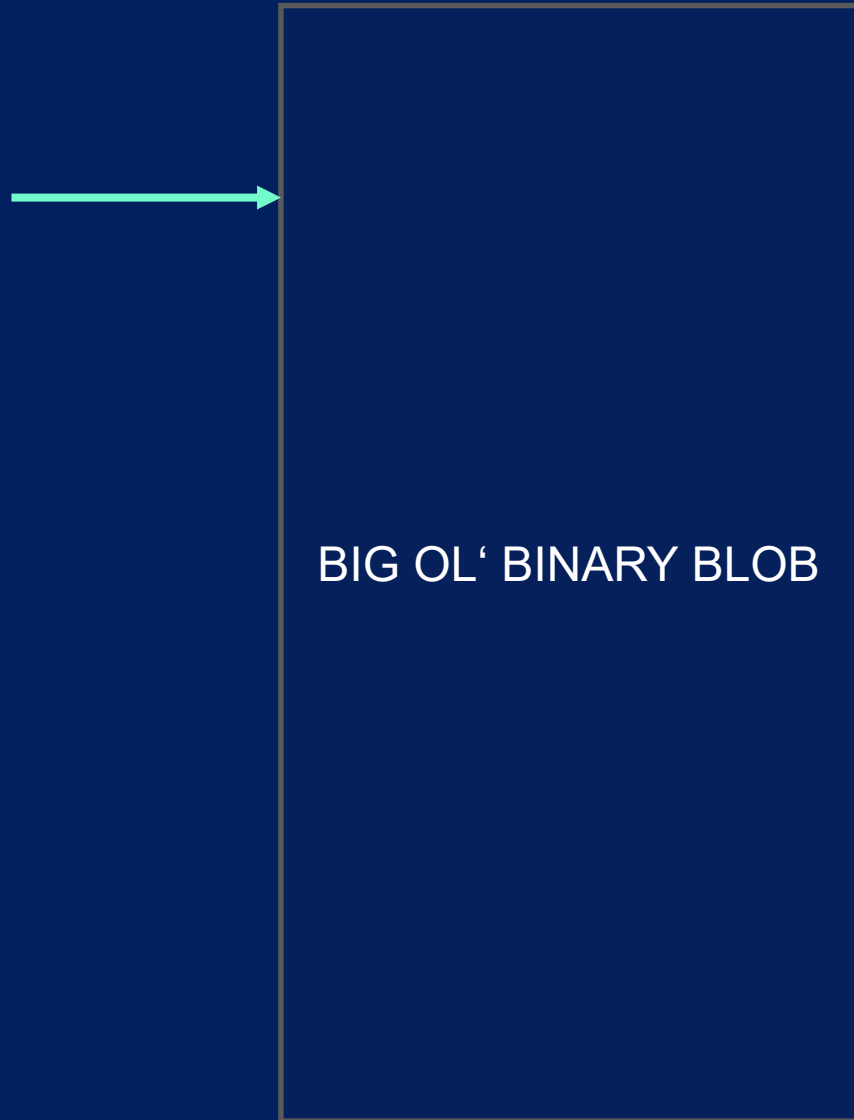


BIG OL' BINARY BLOB

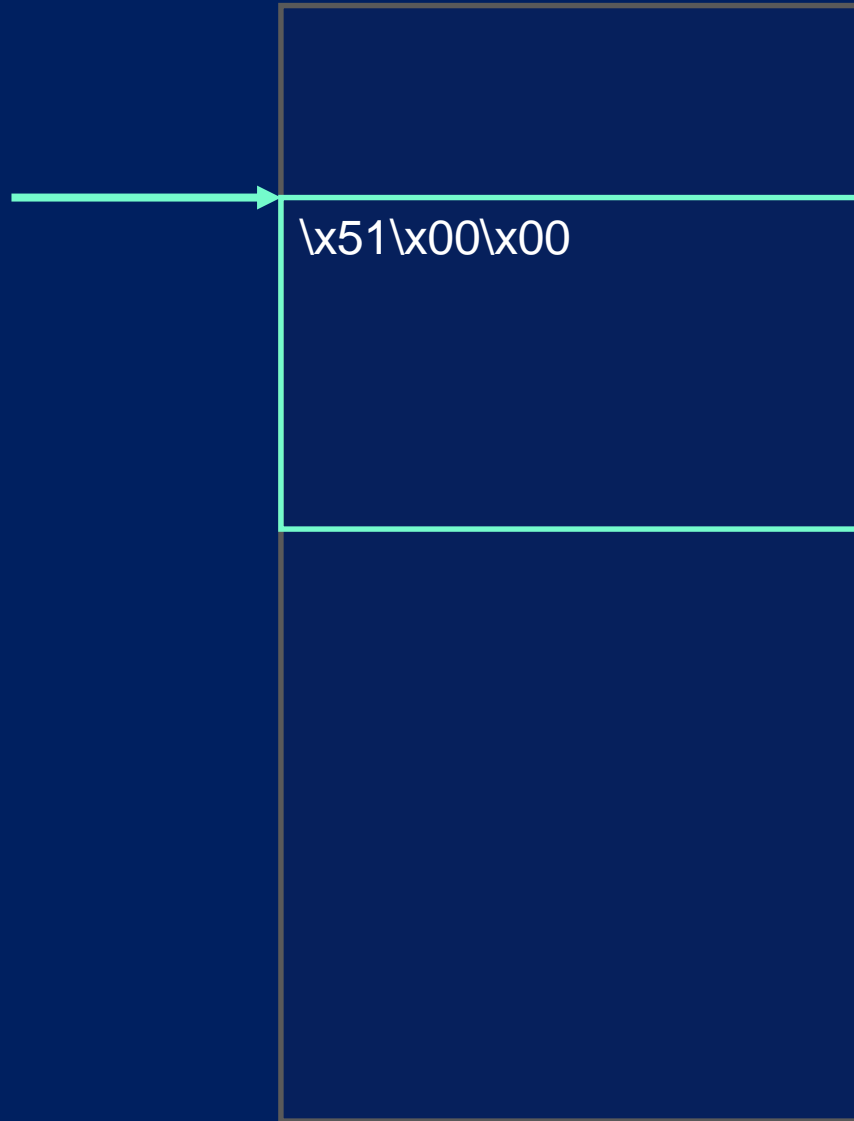
OBJECTIVE 1 : ACCURACY



OBJECTIVE 1 : ACCURACY



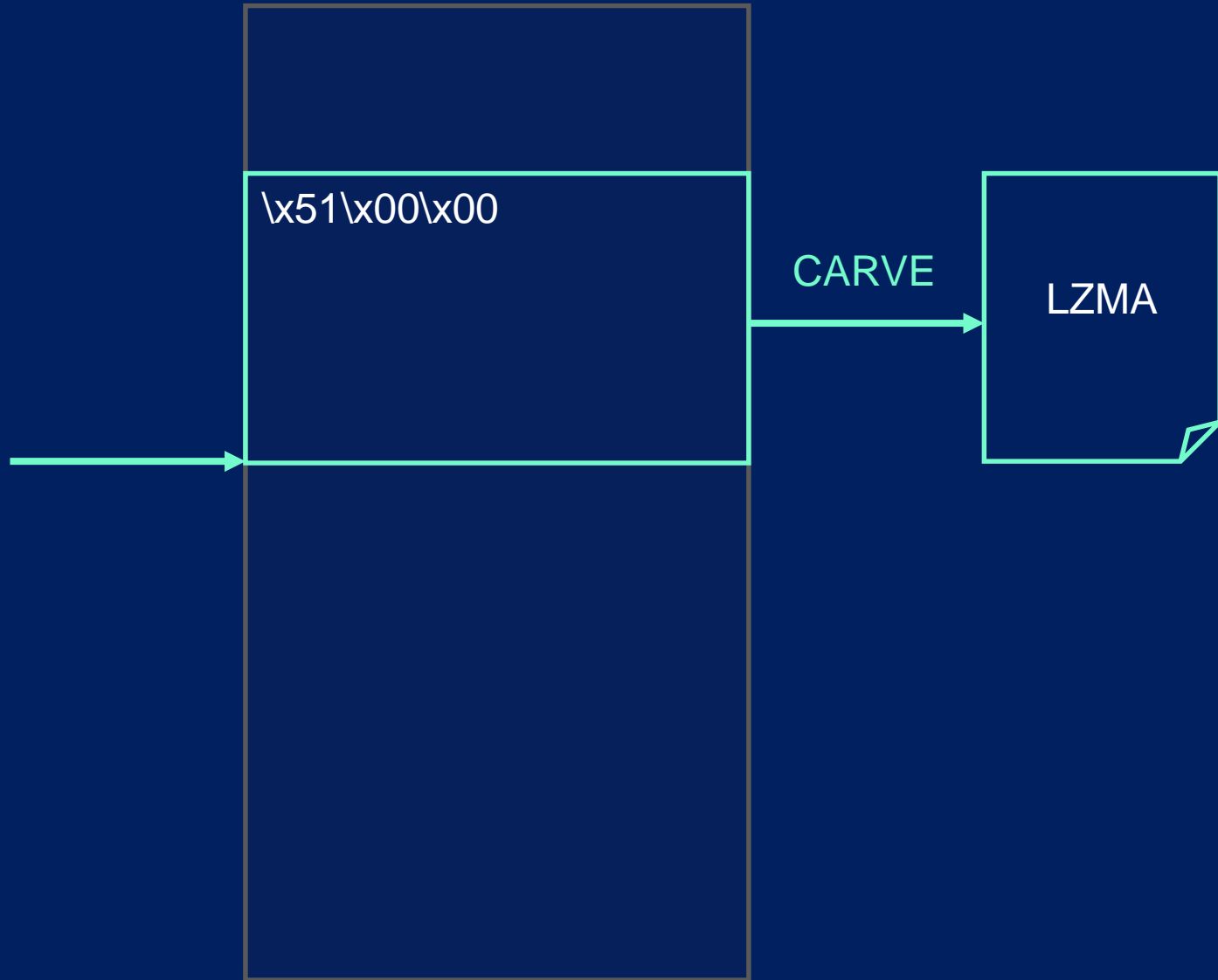
OBJECTIVE 1 : ACCURACY



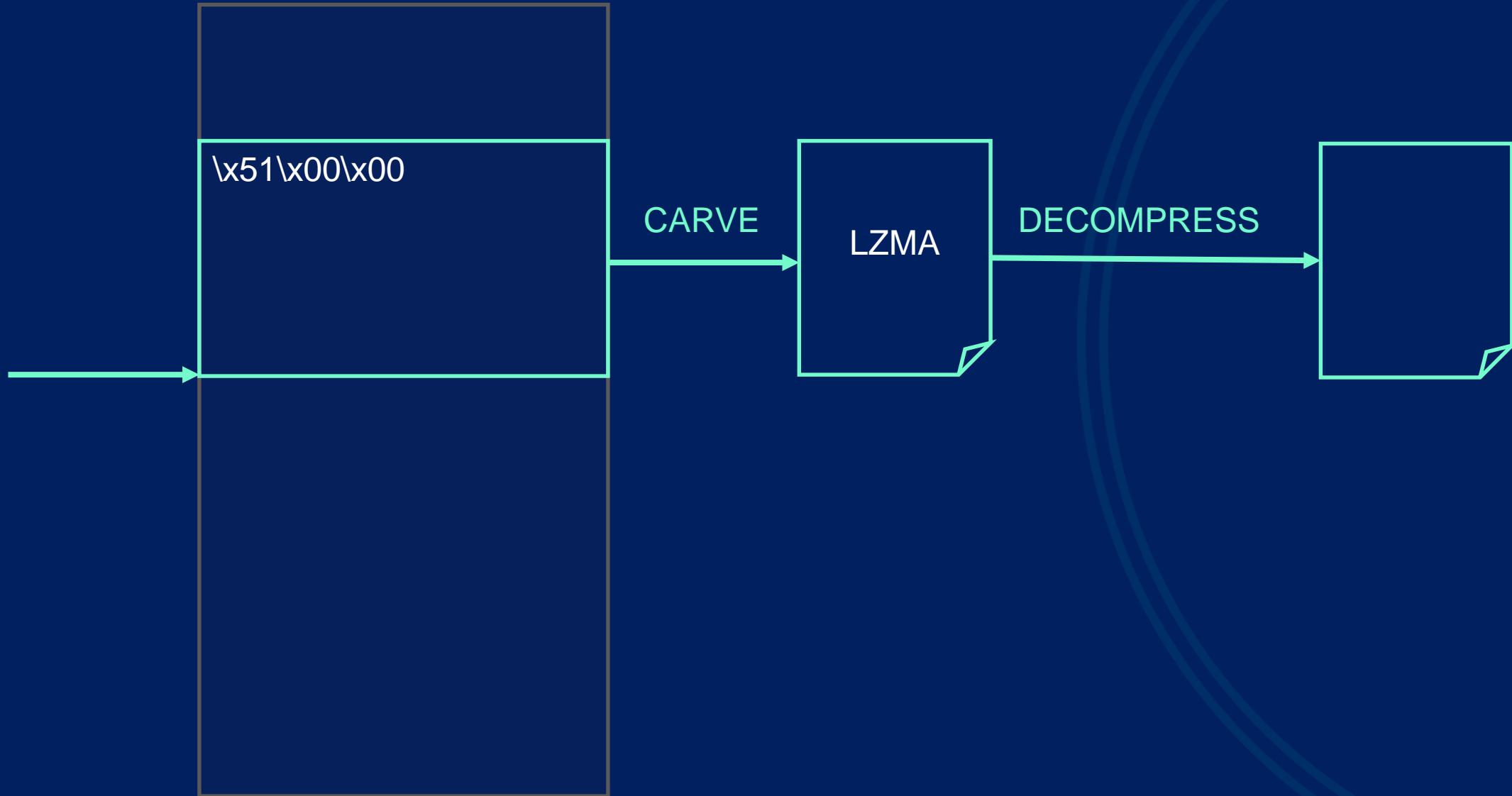
OBJECTIVE 1 : ACCURACY



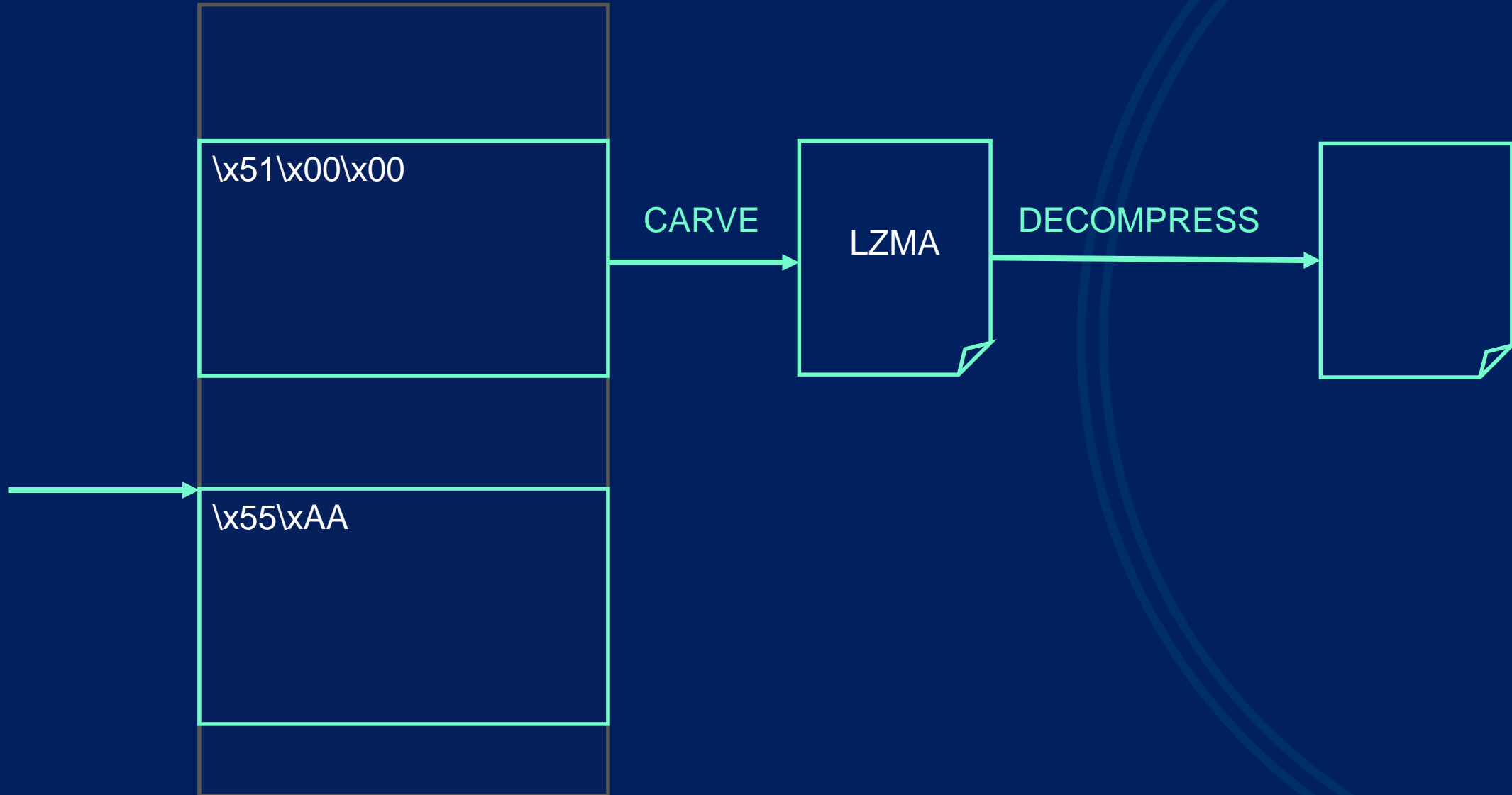
OBJECTIVE 1 : ACCURACY



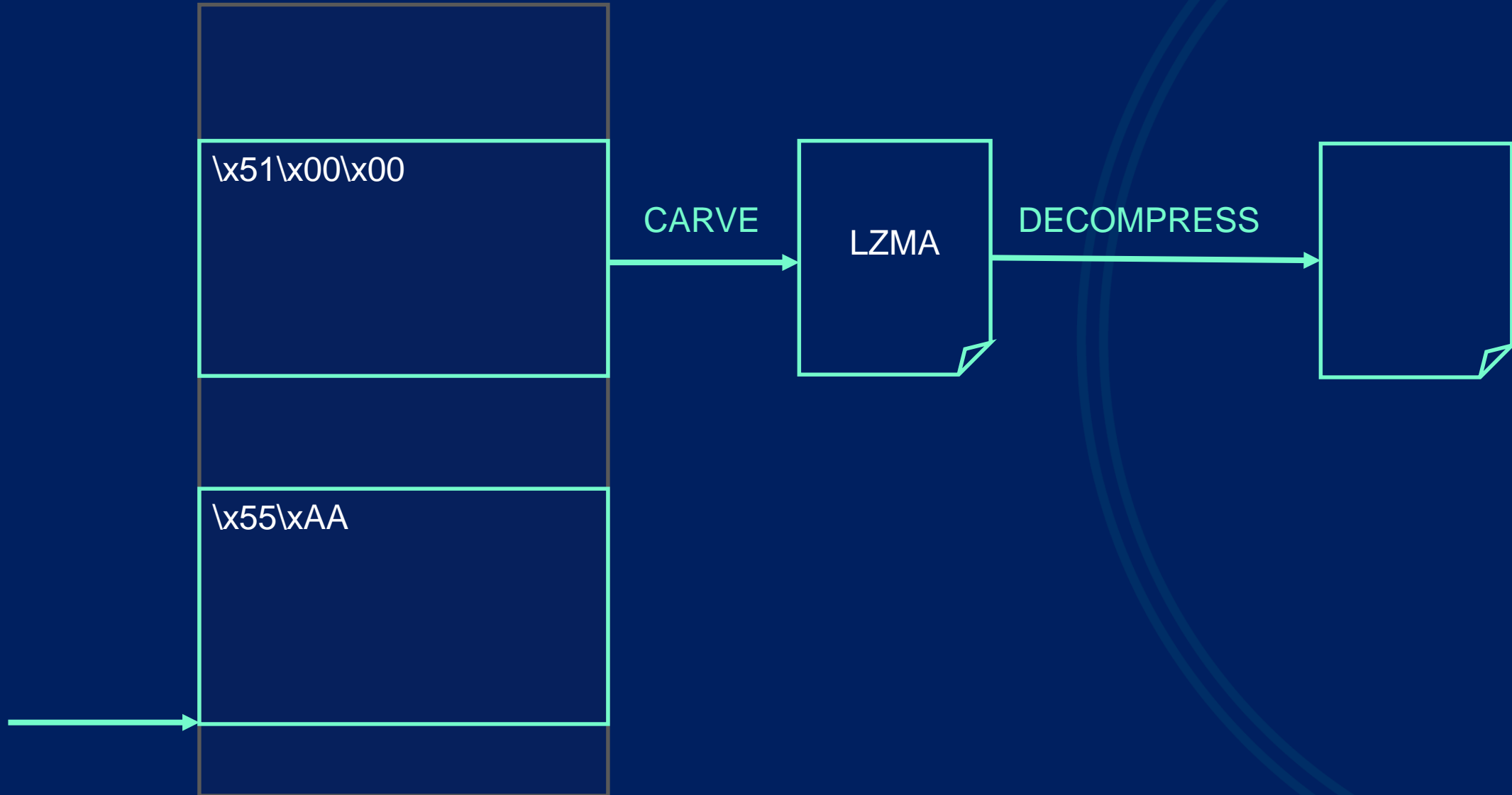
OBJECTIVE 1 : ACCURACY



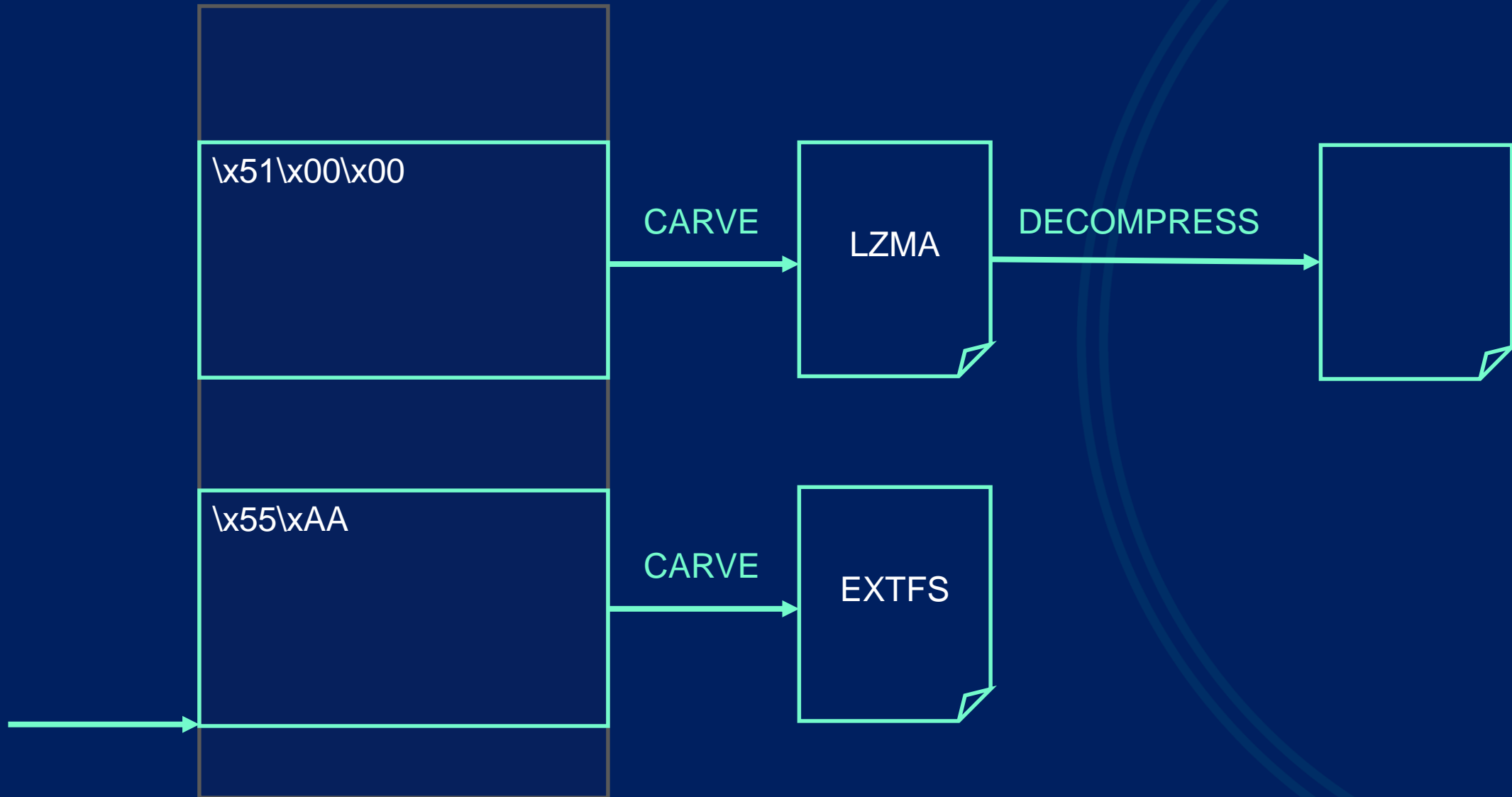
OBJECTIVE 1 : ACCURACY



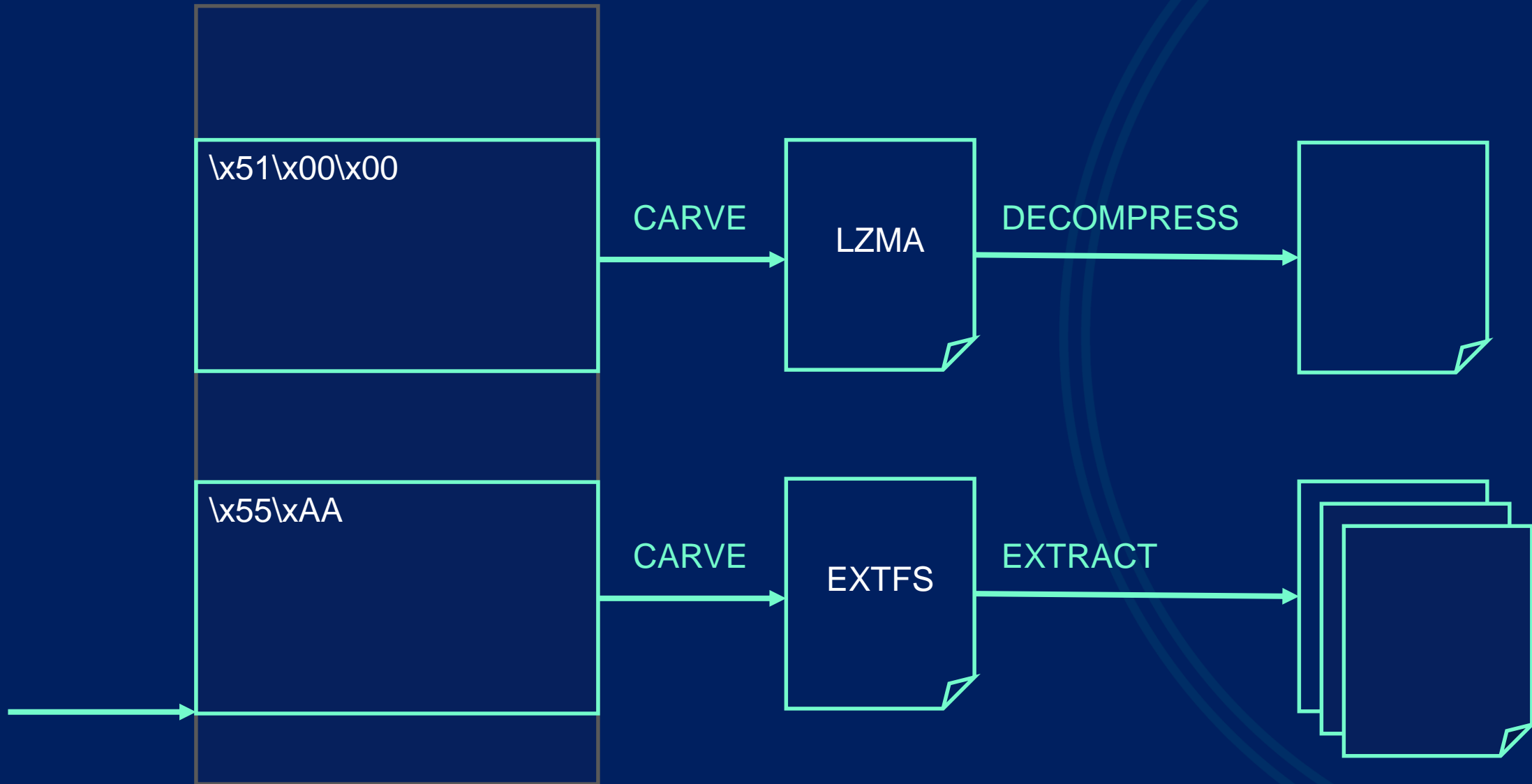
OBJECTIVE 1 : ACCURACY



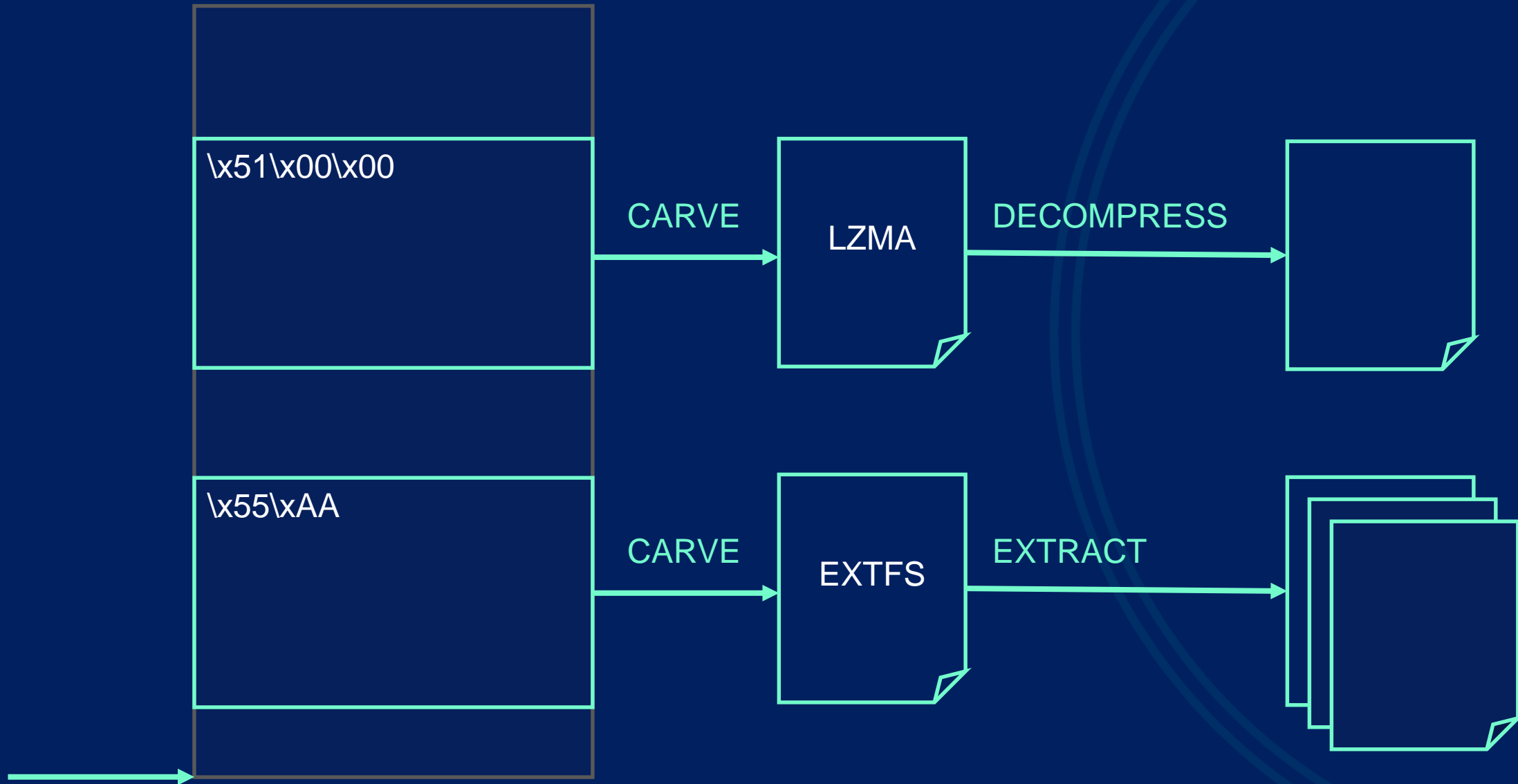
OBJECTIVE 1 : ACCURACY



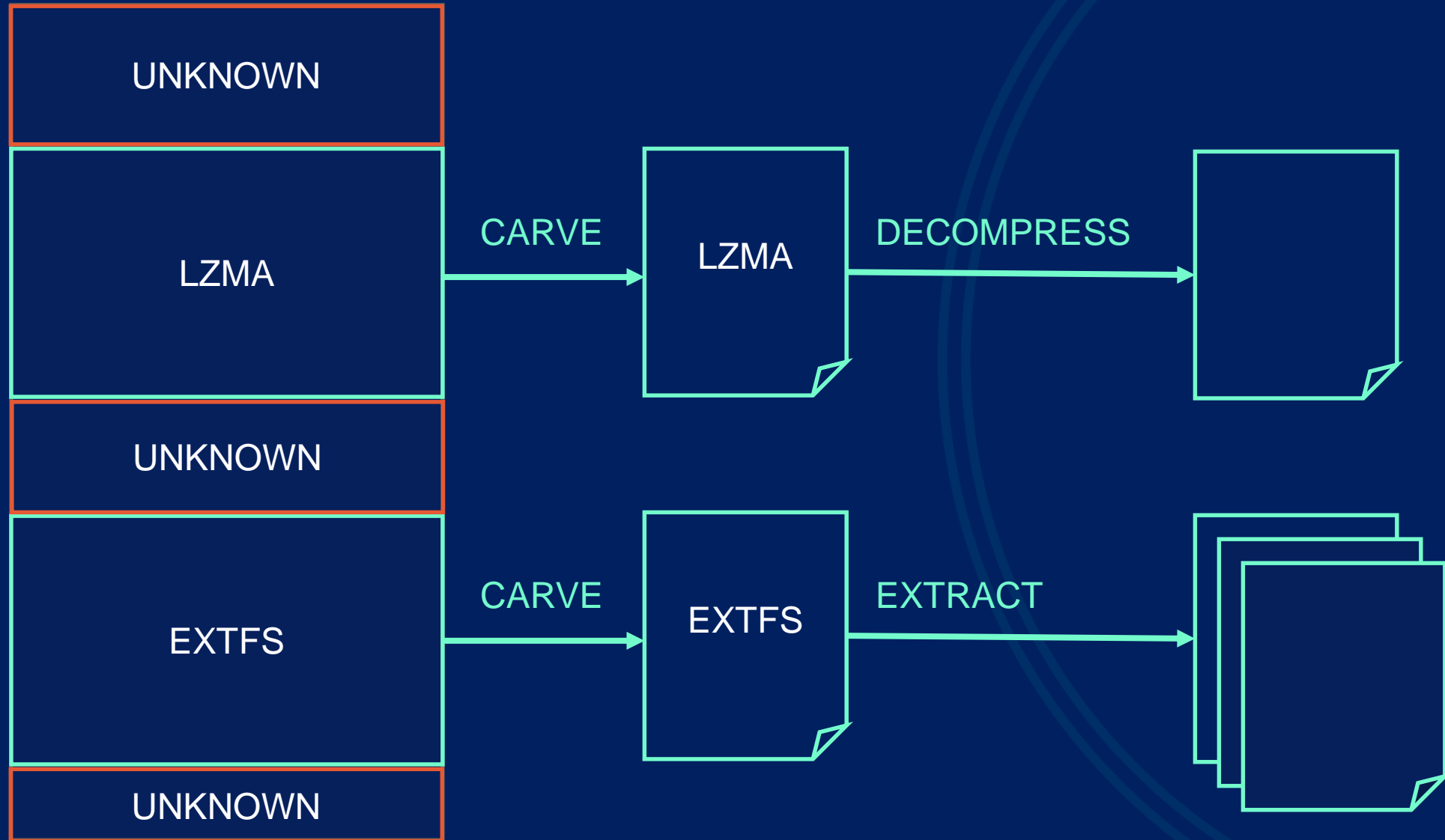
OBJECTIVE 1 : ACCURACY



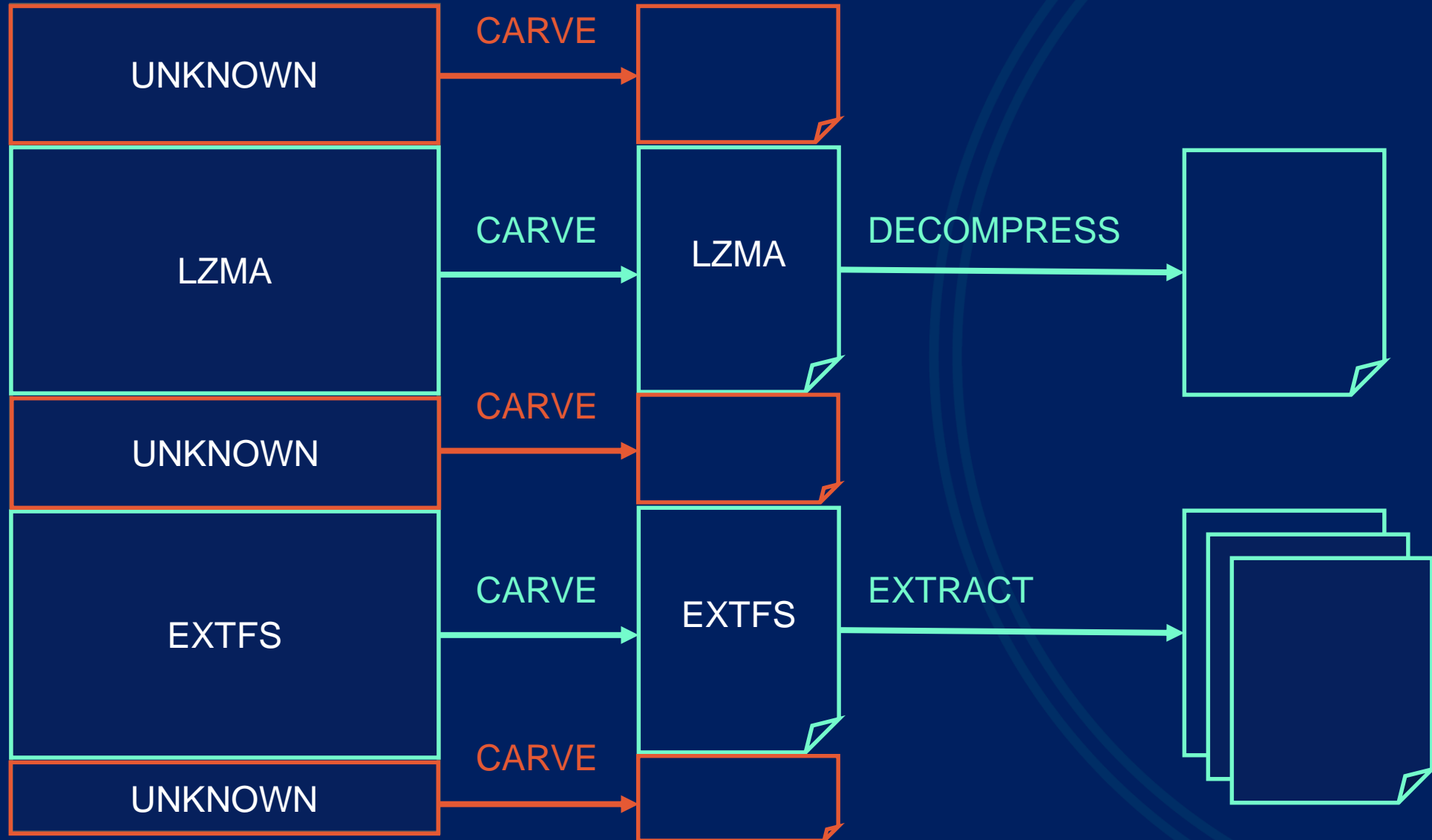
OBJECTIVE 1 : ACCURACY



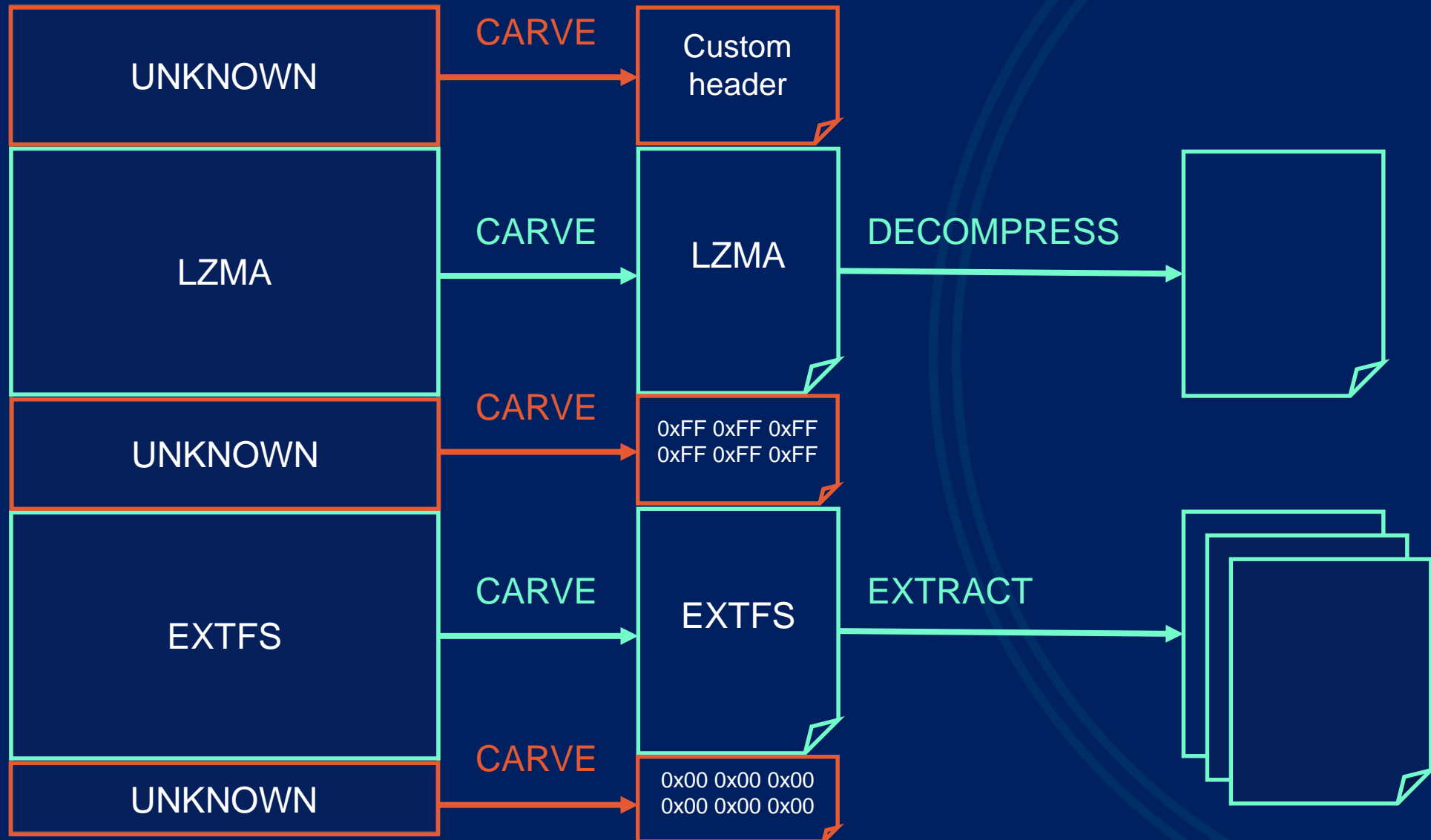
OBJECTIVE 1 : ACCURACY



OBJECTIVE 1 : ACCURACY



OBJECTIVE 1 : ACCURACY



BEING ACCURATE IN A WORLD OF EMBEDDED VENDORS WITH GREAT FIRMWARE IDEAS™

AKA HOW TO LOSE YOUR MIND IN 6 TO 8 MONTHS

KEYSTONE

THE WONDERFUL WORLD OF VENDOR FORMATS

	v2							
LE	hsqs							
BE	sqsh							

THE WONDERFUL WORLD OF VENDOR FORMATS

	v2	v3						
LE	hsqs	hsqs						
BE	sqsh	sqsh						

THE WONDERFUL WORLD OF VENDOR FORMATS

	v2	v3	v4					
LE	hsqs	hsqs	hsqs					
BE	sqsh	sqsh	sqsh					

THE WONDERFUL WORLD OF VENDOR FORMATS

	v2	v3	v4	DDWRT v3				
LE	hsqs	hsqs	hsqs	hsqt				
BE	sqsh	sqsh	sqsh	tqsh				

THE WONDERFUL WORLD OF VENDOR FORMATS

	v2	v3	v4	DDWRT v3	Broadcom v3			
LE	hsqs	hsqs	hsqs	hsqt	shsq			
BE	sqsh	sqsh	sqsh	tqsh	qshs			

THE WONDERFUL WORLD OF VENDOR FORMATS

	v2	v3	v4	DDWRT v3	Broadcom v3	??? v3		
LE	hsqs	hsqs	hsqs	hsqt	shsq	zlqs		
BE	sqsh	sqsh	sqsh	tqsh	qshs	sqlz		

THE WONDERFUL WORLD OF VENDOR FORMATS

	v2	v3	v4	DDWRT v3	Broadcom v3	??? v3	AVM	
LE	hsqs	hsqs	hsqs	hsqt	shsq	zlqs	Header in BE, but chunks in LE.	
BE	sqsh	sqsh	sqsh	tqsh	qshs	sqlz		

THE WONDERFUL WORLD OF VENDOR FORMATS

	v2	v3	v4	DDWRT v3	Broadcom v3	??? v3	AVM	Netgear
LE	hsqs	hsqs	hsqs	hsqt	shsq	zlqs	Header in BE, but chunks in LE.	Let's use a non standard LZMA + XZ compression !
BE	sqsh	sqsh	sqsh	tqsh	qshs	sqlz		

THE WONDERFUL WORLD OF VENDOR FORMATS

	v2	v3	v4	DDWRT v3	Broadcom v3	??? v3	AVM	Netgear
LE	hsqs	hsqs	hsqs	hsqt	shsq	zlqs	Header in BE, but chunks in LE.	Let's use a non standard LZMA + XZ compression !
BE	sqsh	sqsh	sqsh	tqsh	qshs	sqlz		

THE WONDERFUL WORLD OF VENDOR FORMATS

```
computed_crc=0x3af2f56e header_crc=0x6ef5f23a
```



```
commit 32320fdb88bc930c0b2bc2935ada8bff5fdc9042 (394-disable-cramfs-crc)
```

```
Author: Quentin Kaiser <quentin.kaiser@onekey.com>
```

```
Date: Thu Jun 9 16:17:55 2022 +0200
```

Improve cramfs CRC checking.

We observed a sample from Cisco where the CRC value was stored in a different endianness. This is non standard but we decided to support it anyway.

We also added a check for old cramfs format. If it's an old format, the CRC check always returns true because it's not supported.

More information about CRC validation and old format:
<https://github.com/secularbird/cramfs/blob/master/cramfsck.c>

OBJECTIVE 2 : SECURITY

Extraction Attacks	Stability
Privileges	Dependencies

OBJECTIVE 2 : SECURITY

Extraction Attacks	Stability
Privileges	Dependencies

OBJECTIVE 2 : SECURITY

Extraction Attacks	Stability
Privileges	Dependencies

OBJECTIVE 2 : SECURITY

Extraction Attacks	Stability
Privileges	Dependencies

OBJECTIVE 2 : SECURITY

Extraction Attacks	Stability
Privileges	Dependencies

OBJECTIVE 2 : SECURITY

Extraction Attacks

Stability

Dependencies

Privileges

Audited third party extraction tools we rely on.

Found **path traversals** vulnerabilities in:

- jefferson (fixed)
- ubi-reader (fixed)
- Yaffshiv (fixed in our fork)
- binwalk (fixed in our fork)

And a few other logic bugs related to inode handling (looking at you Stefan 😊).

OBJECTIVE 2 : SECURITY

```
▼ ↕ 25 ■■■■ src/scripts/jefferson 📄

54 + def is_safe_path(basedir, path, follow_symlinks=True):
55 +     if follow_symlinks:
56 +         matchpath = os.path.realpath(path)
57 +     else:
58 +         matchpath = os.path.abspath(path)
59 +     return basedir == os.path.commonpath((basedir, matchpath))

55 60
56 61 cstruct.typedef("uint8", "uint8_t")
57 62 cstruct.typedef("uint16", "jint16_t")

⋮
↓
↑
⋮

@@ -474,20 +479,32 @@ def dump_fs(fs, target):

474 479     node_names.append(dirent.name.decode())
475 480     path = "/".join(node_names)
476 481
477 -     target_path = os.path.join(os.getcwd(), target, path)
482 +     target_path = os.path.realpath(os.path.join(os.getcwd(), target, path))
483 +
484 +     if not is_safe_path(target, target_path):
485 +         print(f"Path traversal attempt to {target_path}, discarding.")
486 +         continue
487 +
```

OBJECTIVE 2 : SECURITY

```
ubireader/ubifs/output.py

29 + def is_safe_path(basedir, path, follow_symlinks=True):
30 +     if follow_symlinks:
31 +         matchpath = os.path.realpath(path)
32 +     else:
33 +         matchpath = os.path.abspath(path)
34 +     return basedir == os.path.commonpath((basedir, matchpath))

29 35
30 36 def extract_files(ubifs, out_path, perms=False):
31 37     """Extract UBIFS contents to_path/

@@ -60,7 +66,11 @@ def extract_dents(ubifs, inodes, dent_node, path='', perms=False):

60 66
61 67     inode = inodes[dent_node.inum]
62 68     dent_path = os.path.join(path, dent_node.name)
63 -
69 +
70 +     if not is_safe_path(path, dent_path):
71 +         log(extract_dents, 'path traversal attempt: %s, discarding' % (dent_path))
72 +         return
73 +
```

OBJECTIVE 2 : SECURITY

```
src/yaffshiv

8      13      class Compat(object):
9      14          ...
10     15          Python2/3 compatability methods.

@@ -607,13 +612,14 @@ class YAFFSExtractor(YAFFS):
607     612          for (entry_id, file_path) in Compat.iterator(self.file_paths):
608     613              entry = self.file_entries[entry_id]
609     614              if file_path and int(entry.yaffs_obj_type) == self.YAFFS_OBJECT_TYPE_DIRECTORY:
615     615 +                  file_path = os.path.join(outdir, file_path)
616     616 +
617     617              # Check the file name for possible path traversal attacks
618     618 -                  if b'..' in file_path:
619     619 +                  if not is_safe_path(outdir, file_path):
620     619                      sys.stderr.write("Warning: Refusing to create directory '%s': possible path traversal\n" % file_path)
621     620                      continue
622     621                      try:
623     622 -                          file_path = os.path.join(outdir, file_path)
624     623                          os.makedirs(file_path)
625     624                          self._set_mode_owner(file_path, entry)
626     625                          dir_count += 1
```

OBJECTIVE 2 : SECURITY



Quentin Kaiser Oct 17th at 6:46 AM

Only real python developers can identify the bug in this code. Will you find it ?

IMG_1896 ▼

```
try:
    with PFS(fname) as fs:
        # The end of PFS meta data is the start of the actual data
        data = binwalk.core.common.BlockFile(fname, 'rb')
        data.seek(fs.get_end_of_meta_data())
        for entry in fs.entries():
            outfile_path = os.path.join(out_dir, entry.fname)
            if not outfile_path.startswith(out_dir):
                binwalk.core.common.warning("Urpfs extractor detected directory tra"
            else:
                self._create_dir_from_fname(outfile_path)
                outfile = binwalk.core.common.BlockFile(outfile_path, 'wb')
                outfile.write(data.read(entry.fsize))
                outfile.close()

        data.close()
except KeyboardInterrupt as e:
    raise e
except Exception as e:
    return False
```



1



OBJECTIVE 2 : SECURITY

```
try:
    with PFS(fname) as fs:
        # The end of PFS meta data is the start of the actual data
        data = binwalk.core.common.BlockFile(fname, 'rb')
        data.seek(fs.get_end_of_meta_data())
        for entry in fs.entries():
            outfile_path = os.path.join(out_dir, entry.fname)
            if not outfile_path.startswith(out_dir):
                binwalk.core.common.warning("Unpfs extractor detected directory traversal")
            else:
                self._create_dir_from_fname(outfile_path)
                outfile = binwalk.core.common.BlockFile(outfile_path, 'wb')
                outfile.write(data.read(entry.fsize))
                outfile.close()
        data.close()
except KeyboardInterrupt as e:
    raise e
except Exception as e:
    return False
```


OBJECTIVE 2 : SECURITY

```
src/binwalk/plugins/unpfs.py

@@ -104,7 +104,7 @@ def extractor(self, fname):
    104     104         data = binwalk.core.common.BlockFile(fname, 'rb')
    105     105         data.seek(fs.get_end_of_meta_data())
    106     106         for entry in fs.entries():
    107     -         outfile_path = os.path.join(out_dir, entry.fname)
    107     +         outfile_path = os.path.abspath(os.path.join(out_dir, entry.fname))
    108     108         if not outfile_path.startswith(out_dir):
    109     109             binwalk.core.common.warning("Unpfs extractor detected directory traversal
    110     110         else:
```

OBJECTIVE 2 : SECURITY



Quentin Kaiser

@qkaiser



If I got my ICC embedding right, this file should be a zero day.



7:24 PM · Nov 1, 2022 · Twitter Web App

1 Retweet 1 Quote Tweet 3 Likes

OBJECTIVE 2 : SECURITY

Extraction Attacks

Stability

Dependencies

Privileges

- Fuzz testing of unblob code base.
- Found and fixed 19 bugs so far.
- Logic bugs on malformed / corrupted files mostly.

OBJECTIVE 2 : SECURITY

Extraction Attacks

Stability

Dependencies

Privileges

<input type="checkbox"/> 0 Open <input checked="" type="checkbox"/> 19 Closed		Author	Label	Projects	Milestones	Assignee	Sort
<input type="checkbox"/>	<input checked="" type="checkbox"/> infinite loop in zlib handler makes unblob hang	bug	format:compression	fuzzing	1		
#441 by QKaiser was closed on Sep 27							
<input type="checkbox"/>	<input checked="" type="checkbox"/> Unhandled zlib error in gzip handler when parsing corrupted gzip files	bug	format:compression	fuzzing	1		
#200 by QKaiser was closed on Jan 27 v1.0 - extraction							
<input type="checkbox"/>	<input checked="" type="checkbox"/> Uncaught PermissionError in unblob on malformed JFFS2 filesystems	bug	format:filesystem	fuzzing	1		4
#190 by QKaiser was closed on Jan 28 v1.0 - extraction							
<input type="checkbox"/>	<input checked="" type="checkbox"/> Uncaught ValueError in rar file handler on malformed rar archives	bug	format:archive	fuzzing	1		
#189 by QKaiser was closed on Jan 26 v1.0 - extraction							
<input type="checkbox"/>	<input checked="" type="checkbox"/> Uncaught ReadError in tar file handler on malformed tar archive	bug	format:archive	fuzzing	1		
#188 by QKaiser was closed on Jan 25 v1.0 - extraction							
<input type="checkbox"/>	<input checked="" type="checkbox"/> Uncaught InvalidHeaderError in tar file handler on malformed tar files	bug	format:archive	fuzzing	1		
#187 by QKaiser was closed on Jan 25 v1.0 - extraction							
<input type="checkbox"/>	<input checked="" type="checkbox"/> Uncaught UnicodeDecodeError in unblob extraction on malformed CAB files	bug	format:archive	fuzzing	1		
#186 by QKaiser was closed on Jan 28 v1.0 - extraction							
<input type="checkbox"/>	<input checked="" type="checkbox"/> Negative seek in ZIP handler on malformed file	bug	format:archive	fuzzing	1		
#185 by QKaiser was closed on Jan 26 v1.0 - extraction							
<input type="checkbox"/>	<input checked="" type="checkbox"/> Unhandled UnicodeEncodeError in unblob through corrupted CAB files	bug	format:archive	fuzzing	1		1
#184 by QKaiser was closed on Jan 28 v1.0 - extraction							
<input type="checkbox"/>	<input checked="" type="checkbox"/> Unhandled BadZipFile exception in ZIP handler when parsing corrupted ZIP files	bug	format:archive	fuzzing	1		
#183 by QKaiser was closed on Jan 26 v1.0 - extraction							
<input type="checkbox"/>	<input checked="" type="checkbox"/> Unhandled UnicodeDecodeError in ZIP handler when parsing corrupted ZIP files	bug	format:archive	fuzzing	1		
#182 by QKaiser was closed on Jan 26 v1.0 - extraction							
<input type="checkbox"/>	<input checked="" type="checkbox"/> LZ4 triggers ValueError on malformed files	bug	format:compression	fuzzing	2		
#177 by QKaiser was closed on Jan 27 v1.0 - extraction							

OBJECTIVE 2 : SECURITY

Extraction Attacks

Stability

Dependencies

Privileges

- All our dependencies are documented and **locked** with poetry.
- Even better if you use our Nix build.

OBJECTIVE 2 : SECURITY

Extraction Attacks

Stability

Dependencies

Privileges

- No elevated privileges required.
- Opinionated decision, we can't create special files.

OBJECTIVE 3 : EXTENSIBILITY

A Brief History of Metasploit

Metasploit was originally developed and conceived by HD Moore while he was employed by a security firm. When HD realized that he was spending most of his time validating and sanitizing public exploit code, he began to create a flexible and maintainable framework for the creation and development of exploits. He released his first edition of the Perl-based Metasploit in October 2003 with a total of 11 exploits.




With the help of Spoonm, HD released a total rewrite of the project, Metasploit 2.0, in April 2004. This version included 19 exploits and over 27 payloads. Shortly after this release, Matt Miller (Skape) joined the Metasploit development team, and as the project gained popularity, the Metasploit Framework received ...


OBJECTIVE 3 : EXTENSIBILITY


GitHub Gist

Search...


All gists Back to GitHub

 nstarke / [extract-netgear-chk-firmware.md](#)

 Star


4


 Fork


2


Created 4 years ago • Report abuse

<> Code


 Revisions 1

 Stars 4

 Forks 2

 Download ZIP

Extract Netgear .chk Firmware

 [extract-netgear-chk-firmware.md](#)

Raw

Extract Netgear .chk Firmware

I recently ran into a situation where `binwalk -M -e $FIRMWARE` failed me. This was for a Netgear firmware image that ended in a `.chk` extension.

The firmware file name was `R7960P-V1.0.1.34_1.0.20.chk`.

This is the output when I ran `binwalk R7960P-V1.0.1.34_1.0.20.chk`:

OBJECTIVE 3 : EXTENSIBILITY

The screenshot shows a GitHub repository page for 'HaToan / Decrypt-Firmware-Hikvision'. The repository is public and has 1 watch, 2 forks, and 7 stars. The 'Code' tab is selected, showing a list of files: README.md, dec_hik.py, and digicap.dav, all uploaded 14 months ago. The README.md file is expanded, showing the title 'Decrypt Firmware Hikvision' and the subtitle 'Decrypt E3S'. The right sidebar contains the 'About' section, which states 'No description, website, or topics provided.', and the 'Releases' section, which states 'No releases published'.

HaToan / Decrypt-Firmware-Hikvision Public

Watch 1 Fork 2 Star 7

Code Issues Pull requests Actions Projects Security Insights

master Go to file Add file Code About

HaToan Decrypt firmware E3S on Sep 24, 2021 1

README.md	Decrypt firmware E3S	14 months ago
dec_hik.py	Decrypt firmware E3S	14 months ago
digicap.dav	Decrypt firmware E3S	14 months ago

README.md

Decrypt Firmware Hikvision

Decrypt E3S

No description, website, or topics provided.

Readme 7 stars 1 watching 2 forks

Releases No releases published

Packages

OBJECTIVE 3 : EXTENSIBILITY

krolinventions / draytools Public archive Watch 7 Fork 53 Star 55

[Code](#) [Issues 3](#) [Pull requests 1](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#)

master Go to file Code About

krolinventions Improve bruteforcing by not relying on s... ... on Jun 21, 2012 25

CHANGELOG	Fixed some null-terminated credentials (rare...	11 years ago
COPYING	first public release	11 years ago
CREDITS	refactoring & new features, now v0.3	11 years ago
INSTALL	refactoring & new features, now v0.3	11 years ago
README	added explicit "admin" username for master ...	11 years ago
VERSION	Fixed some null-terminated credentials (rare...	11 years ago
draytools.py	Improve bruteforcing by not relying on smar...	11 years ago
pydelzo.py	cfg now ok for 2700 and 2800	11 years ago

README

About

DrayTek Vigor password recovery, config & firmware tools - NOT MAINTAINED, CHECK

[github.com/gerard-/draytools/net...](#)

Readme

GPL-3.0 license

55 stars

7 watching

53 forks

Releases

7 tags

OBJECTIVE 3 : EXTENSIBILITY

The screenshot shows the GitHub interface for the repository 'yath/vigor165'. At the top, the repository name is displayed with a 'Public' badge. To the right are buttons for 'Watch' (2), 'Fork' (2), and 'Star' (3). Below this is a navigation bar with links for 'Code', 'Issues' (1), 'Pull requests', 'Actions', 'Projects', 'Security', and 'Insights'. The 'Code' tab is selected. Below the navigation bar, there are buttons for 'main', 'Go to file', 'Add file', and 'Code'. The main content area shows a commit by 'yath' on Oct 30, 2020, with 8 commits. The commit message is 'Write compressed sections as individual files'. Below the commit message is a table of files:

decompress	Write compressed sections as individual files	2 years ago
dump	Add old memory dumping code	2 years ago
README.md	Add notes	2 years ago

Below the table is a section for 'README.md'. The title is 'Vigor165 notes'. The text reads: 'This repository collects some notes on my reverse engineering efforts on the DrayTek Vigor 165.' On the right side of the repository page, there is an 'About' section with the text 'DrayTek Vigor165 stuff'. Below this are links for 'Readme', '3 stars', '2 watching', and '2 forks'. At the bottom right, there is a 'Languages' section with a horizontal bar chart showing the following data:

Language	Percentage
Go	47.3%
Assembly	33.8%
C	17.0%
Makefile	1.9%

OBJECTIVE 3 : EXTENSIBILITY

The screenshot shows the GitHub interface for the repository 'synacktiv/yealink_tools'. At the top, it indicates the repository is 'Public' and shows 6 watchers, 5 forks, and 10 stars. The navigation bar includes links for Code, Issues (1), Pull requests, Actions, Projects, Security, and Insights. Below the navigation bar, there are buttons for 'master' branch, 'Go to file', 'Add file', and 'Code'. The main content area displays a list of files in the repository, including README.md, yaffs.ksy, yaffs_decrypt.py, yaffs_tags.ksy, yealink_crypto.py, yealink_rom.ksy, and yealink_rom_dump.py. The right sidebar contains the 'About' section, which describes the repository as 'Reverse engineering scripts designed for extracting Yealink VOIP upgrade files', and lists statistics: 10 stars, 6 watching, and 5 forks. Below the 'About' section are sections for 'Releases' and 'Packages', both of which show 'No releases published' and 'No packages published' respectively.

synacktiv / yealink_tools Public

Watch 6 Fork 5 Star 10

Code Issues 1 Pull requests Actions Projects Security Insights

master Go to file Add file Code

synacktiv-tp Add README on Sep 4, 2019 2

README.md	Add README	3 years ago
yaffs.ksy	Initial commit	3 years ago
yaffs_decrypt.py	Initial commit	3 years ago
yaffs_tags.ksy	Initial commit	3 years ago
yealink_crypto.py	Initial commit	3 years ago
yealink_rom.ksy	Initial commit	3 years ago
yealink_rom_dump.py	Initial commit	3 years ago

README.md

Yealink firmware reverse engineering

About

Reverse engineering scripts designed for extracting Yealink VOIP upgrade files

Readme

10 stars

6 watching

5 forks

Releases

No releases published

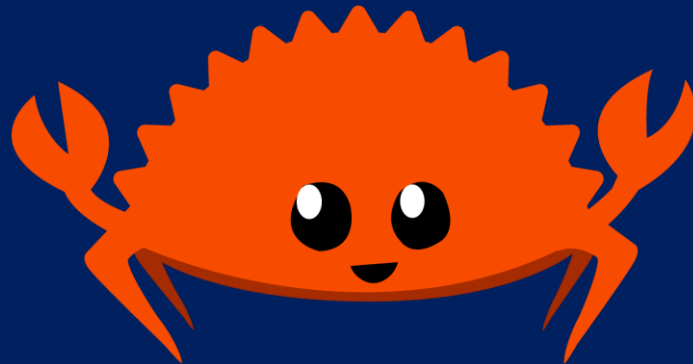
Packages

No packages published


OBJECTIVE 4 : SPEED



Hyperscan



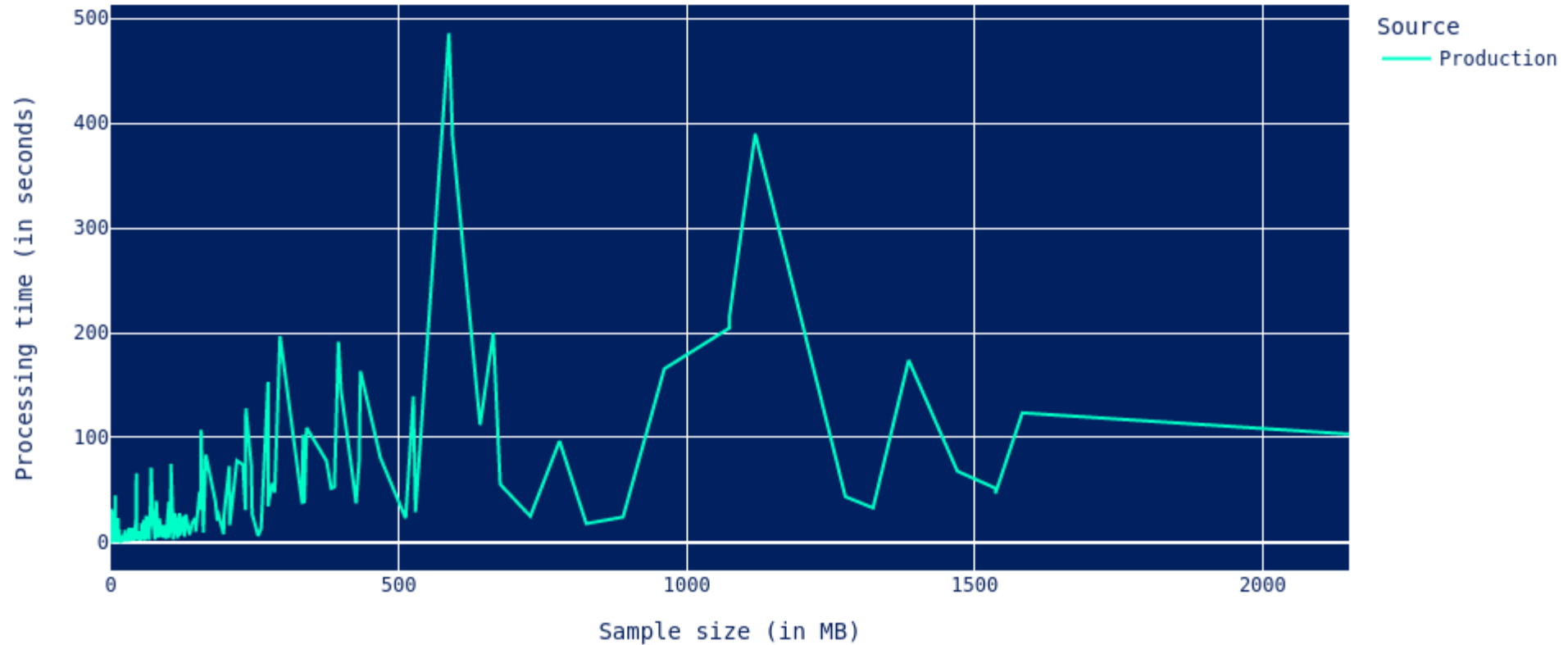
Rust

mmap in
Python
 AskPython

MMAP'ing

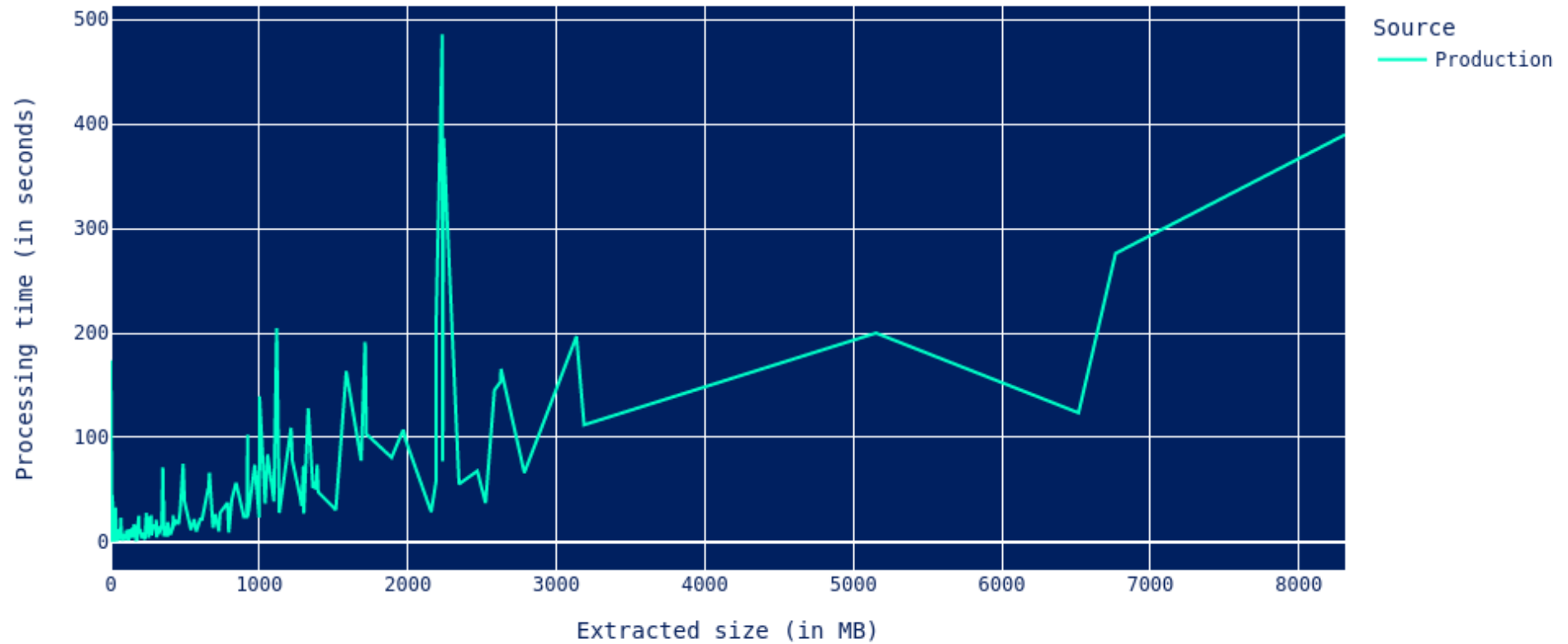
OBJECTIVE 4 : SPEED

Unblob processing time distribution



OBJECTIVE 4 : SPEED

Unblob processing time distribution



The team behind **unblob**



tests00



György Kiss
kissgyorgy



János Kukovecz
kukovecz



Mücahid KIR
mucoze



Quentin Kaiser
QKaiser



Krisztián Fekete
e3krisztian



László Vaskó
vlaci



Marton Illes
martonilles



mzombor
r4bbit-r4

AGENDA

1. Getting Started
2. Extraction, Analysis & Reporting
3. Creating a Format Handler
4. Creating a Format Extractor
5. Q&A

GETTING STARTED

You can start with unblob in 3 different ways:

1. **Docker**
2. **Nix**
3. **Source**

GETTING STARTED (DOCKER)



```
docker run \  
  --rm \  
  --pull always \  
  -v /path/to/extract-dir/on/host:/data/output \  
  -v /path/to/files/on/host:/data/input \  
ghcr.io/onekey-sec/unblob:latest /data/input/path/to/file
```

GETTING STARTED (NIX)



```
nix profile install github:onekey-sec/unblob
```

```
unblob --show-external-dependencies
```

The following executables found installed, which are needed by unblob:

7z	✓
jefferson	✓
lz4	✓
lziprecover	✓
lzop	✓
simg2img	✓
tar	✓
ubireader_extract_files	✓
ubireader_extract_images	✓
unar	✓
unromfs	✓
unsquashfs	✓
yaffshiv	✓

GETTING STARTED (SOURCE)



```
git clone https://github.com/onekey-sec/unblob.git  
cd unblob  
poetry install --no-dev
```

LET'S EXTRACT SOME STUFF !

```
unblob -h
Usage: unblob [OPTIONS] FILE

A tool for getting information out of any kind of binary blob.

You also need these extractor commands to be able to extract the supported
file types: 7z, debugfs, jefferson, lz4, lziprecover, lzop, sasquatch,
sasquatch-v4be, simg2img, ubireader_extract_files, ubireader_extract_images,
unar, yaffshiv, zstd

NOTE: Some older extractors might not be compatible.

Options:
-e, --extract-dir DIRECTORY    Extract the files to this directory. Will be
                                created if doesn't exist.
-f, --force                    Force extraction even if outputs already
                                exist (they are removed).
-d, --depth INTEGER RANGE     Recursion depth. How deep should we extract
                                containers. [default: 10; x>=1]
-n, --entropy-depth INTEGER RANGE
                                Entropy calculation depth. How deep should
                                we calculate entropy for unknown files? 1
                                means input files only, 0 turns it off.
                                [default: 1; x>=0]
-P, --plugins-path PATH        Load plugins from the provided path.
-S, --skip-magic TEXT          Skip processing files with given magic
                                prefix [default: BFLT, JPEG, GIF, PNG,
                                SQLite, compiled Java class, TrueType Font
                                data, PDF document, magic binary file, MS
                                Windows icon resource, PE32+ executable (EFI
                                application)]
-p, --process-num INTEGER RANGE
                                Number of worker processes to process files
                                parallelly. [default: 12; x>=1]
--report PATH                  File to store metadata generated during the
                                extraction process (in JSON format).
-k, --keep-extracted-chunks    Keep extracted chunks
-v, --verbose                  Verbosity level, counting, maximum level: 3
                                (use: -v, -vv, -vvv)
--show-external-dependencies   Shows commands needs to be available for
                                unblob to work properly
-h, --help                    Show this message and exit.
```

EXTRACTORS

Extracting sometimes require external extractors, in some case also non-standard tools that could handle the “creative” extensions of various vendors.

We maintain various forks of extractors:

- jefferson (<https://github.com/onekey-sec/jefferson>)
- sasquatch (<https://github.com/onekey-sec/sasquatch>)
- ubi_reader (https://github.com/onekey-sec/ubi_reader)
- yaffshiv (<https://github.com/onekey-sec/yaffshiv>)

Many other extractors are built into unblob.

FUTURE WORK

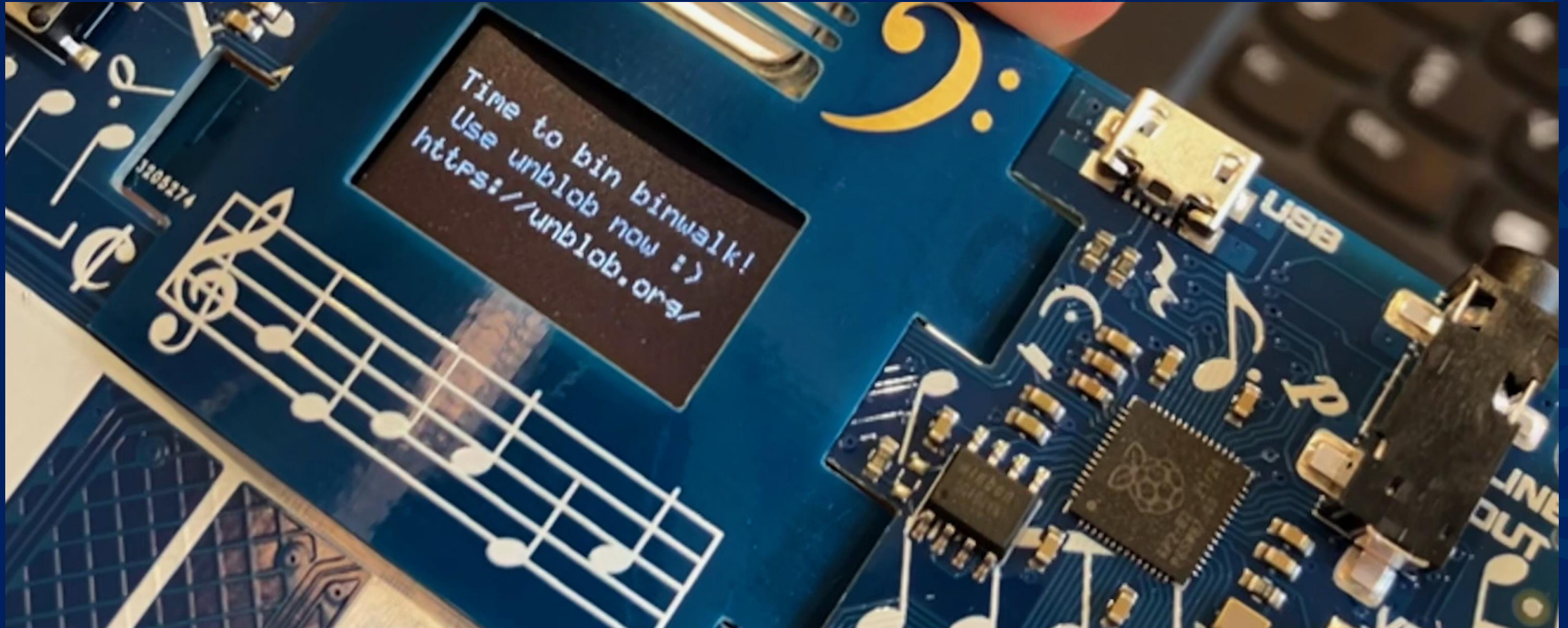
- Clean report of extraction process in console
- Increase meta-data extraction
- Unknown chunks auto-identification (introspection)
- Additional format support

CONTRIBUTE !

- Test it on your weirdest files
- Report bugs
- Request support for new formats
- Submit PR for new handlers and extractors

<https://github.com/onekey-sec/unblob>

CONTRIBUTE !



THANK YOU for your attention.



- Incendiary emails can be sent to research@onekey.com