



Openchange

White Paper

PmatchS: Pattern Matching Solver

baseline) Julien Kerihuel¹

1: j.kerihuel@openchange.org

Abstract

This document describes the PmatchS project.



Openchange

White Paper

Contents

1 Introduction	5
1.1 What is PmatchS?	5
1.2 What PmatchS can't yet do?	5
2 Installing PmatchS	6
2.1 Getting PmatchS	6
2.1.1 CVS	6
2.1.2 OpenChange Website	6
2.2 System requirements	6
2.2.1 Tools requirement	6
2.2.2 Libraries	6
2.3 Installing PmatchS	6
3 Using PmatchS	7
3.1 Command line parameters	7
3.2 PmatchS menu	7
3.2.1 File	7
3.2.2 Highlighted dump	7
3.3 Panels	8
3.3.1 Occurrence panel	8
3.3.2 Offset panel	9
3.3.3 Configuration panel	10
4 Limitations	11
4.1 Static offset algorithm	11
4.1.1 Offset example	11
4.1.2 Data structure example	12
4.2 Multiple files parsing	12
5 Planned features	13
5.1 Algorithm plugin system	13
5.2 XML element structure parsing	13
5.3 Developer API	13
5.4 Highlighted dump enhancements	13
5.5 Configuration file	13
6 Bug Report and features	14



Openchange

White Paper

6.1 Contact 14

1 Introduction

1.1 What is PmatchS?

The PmatchS tool name stands for *Pattern MATCHing Solver* and belongs to the OpenReverse framework. It has been designed to offer facilities when analyzing unknown or undocumented database file formats. PmatchS compares unlimited number of pages¹ between them, extracts the longest occurrences, attributes them a revealing coefficient and display the results using a user-friendly GTK interface.

1.2 What PmatchS can't yet do?

While PmatchS can substitute for painful human actions, it still requires preliminary research effort. The user needs to identify pages he wants to analyse. PmatchS doesn't either implement yet an algorithm that may find approximative occurrences.

¹A page is a blob of data, defined with its offset and size

2 Installing PmatchS

2.1 Getting PmatchS

2.1.1 CVS

This project's SourceForge.net CVS repository can be checked out through anonymous (pserver) CVS with the following instruction set. When prompted for a password for anonymous, simply press the Enter key.

```
cvscvs -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/openchange login
cvscvs -z3 -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/openchange co -P PmatchS
```

2.1.2 OpenChange Website

You can consult the OpenChange website and check for the latest PmatchS release:

```
http://www.openchange.org/?page=pmatchS
```

2.2 System requirements

2.2.1 Tools requirement

The PMK (Pre Make Kit) project, <http://pmk.sourceforge.net>, is an alternative to autotools. Since PMK has been added to the BSD ports and is available under packages for some Linux distribution, we decided to promote this project and use pmk for our tool releases.

2.2.2 Libraries

You need **gtk2** to be installed in order PmatchS to compile.

2.3 Installing PmatchS

```
> cd PmatchS-$version
> pmk
> make
> make install
```

3 Using PmatchS

3.1 Command line parameters

```
PmatchS -f file
        -o [offset1:offset2:offset3:...:offsetn]
        -l {comparing length}
        -h help
```

- **-f** specify the file to be analysed
- **-o** specify a list of offset separated with :
- **-l** defines the comparing length
- **-h** display help

3.2 PmatchS menu

3.2.1 File

New: Creates a new PmatchS session and prompts for a new file to analyse. Any session previously started is discarded.

Open: Open a saved PmatchS session. The session will be valid until modifications have been brought to the examined file.

textbfSave: Saves the PmatchS results into a file.

3.2.2 Highlighted dump

The highlighted dump display a page using HEX/ASCII and highlight occurrences from green to yellow while revealing coefficient tends to 0.

It first asks for a page to display:

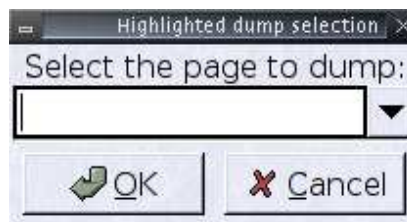


Figure 1: Highlighted dump: Page selection dialog box

Then it loads the highlighted dump. In the example below, the pattern matching process was just executed against two pages so the revealing coefficient can be only be 0 or 100. This is why the only displayed color is green:

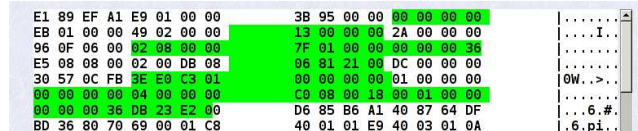


Figure 2: Highlighted dump sample

3.3 Panels

3.3.1 Occurrence panel

The occurrence panel display the occurrences which have been found during the pattern matching process.

Occurrences		Offset	Configuration
Revealance	▲	Offset	Data
100.000000	▼	0x0000000c	length = 0x4
0.000000			value: 00 00 00 00
0.000000			page list: 0x001ea000, 0x001f5000
100.000000	▼	0x00000018	length = 0x4
0.000000			value: 13 00 00 00
0.000000			page list: 0x001ea000, 0x001f5000
100.000000	▶	0x00000024	length = 0xc
100.000000	▶	0x00000038	length = 0x4
100.000000	▶	0x00000044	length = 0x20

Figure 3: Sample occurrences panel results

- The revealance column tells how present is the occurrence in the different pages we compared. It uses a coefficient from 0 to 100. The 0.000000 you can find under the occurrence has no meaning. It's just a bug due to the revealance data representation.

- Since the implemented algorithm uses *static offset*², the offset column tells you where the occurrence is located on the different pages.
- The data column details the occurrence:
 1. The occurrence length
 2. The occurrence value
 3. The list of pages where the occurrence has been found.

3.3.2 Offset panel

This panel manages all the offsets used in the pattern matching process:

- Add an offset: Type the offset in the text field and press the plus button.
- Remove an offset: Select the offset in the list, it will updates the offset text field. Next press the less button.

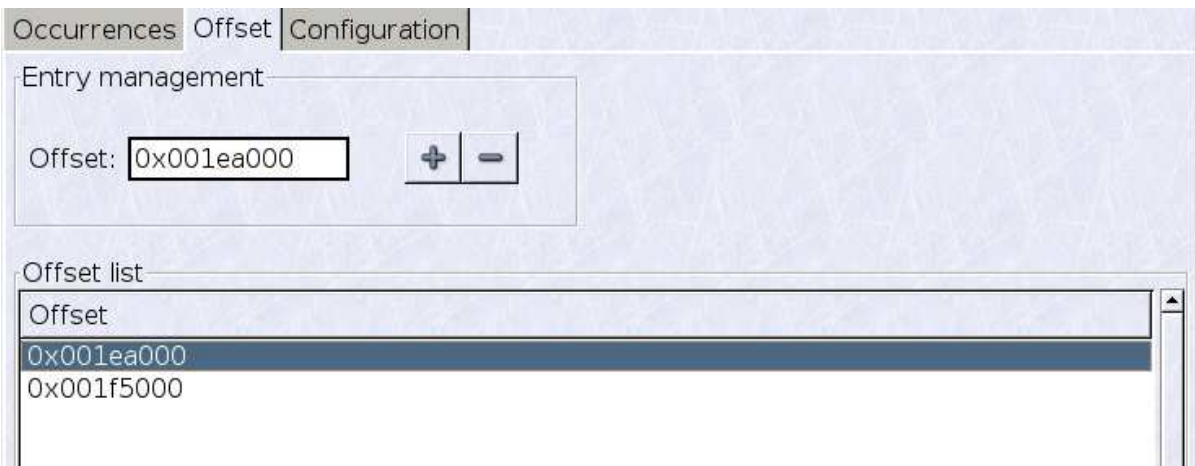


Figure 4: Offset panel

NOTE: The length associated with the offset is unique for all and can be set in the configuration panel. *less* picture.

²Please refer to the Section Limitations for further information

3.3.3 Configuration panel

The configuration panel doesn't contain much information yet. It is only used now to specify the pages length.

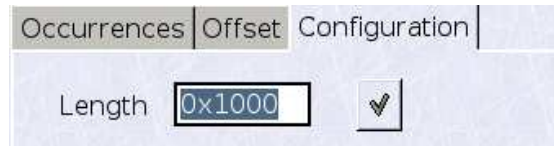


Figure 5: Configuration panel

4 Limitations

4.1 Static offset algorithm

4.1.1 Offset example

When PmatchS compares two pages for occurrences, it uses the same offset on both pages to perform the search. For example, if we take the following *test* file:

```
file test:
 50 6d 61 74 63 68 53 00      00 00 00 00 00 00 00 00      | PmatchS. ....
 00 50 6d 61 74 63 68 53      00 00 00 00 00 00 00 00      | .PmatchS ....
 53 6f 6d 65 20 6f 74 68      65 72 20 64 61 74 61 00      | Some other data.
```

We search through *test* file using 16 bytes length pages; page1 starting at offset 0x0 and page2 starting at offset 0x10

```
./PmatchS -f test -o 0x0:0x10 -l 16

page1 (offset = 0x0, length = 0xF):
 50 6d 61 74 63 68 53 00      00 00 00 00 00 00 00 00      | PmatchS. ....

page2 (offset = 0x10, length = 0xF):
 00 50 6d 61 74 63 68 53      00 00 00 00 00 00 00 00      | .PmatchS ....

Results:
  revealance 100%: 00 00 00 00 00 00 00 00
```

When we would have expected PmatchS to find the string *PmatchS*, it only finds *00 00 00 00 00 00 00 00* at offset 0x8 and 0x18. We should have given PmatchS the following arguments so it could also match the *PmatchS* string:

```
./PmatchS -f test -o 0x0:0x11 -l 16

Results:
  revealance 100%: 50 6d 61 74 63 68 53 00 00 00 00 00 00 00
```

4.1.2 Data structure example

NOTE: In the examples below, we assume `element_1`, `element_2` and `element_3` has the same value on different pages.

While this algorithm can sometimes fit data structure with fixed length such as:

```
typedef struct test {
    uint32_t    element_1;
    uint16_t    element_2;
    char        data[16];
    uint32_t    element_3;
};
```

PmatchS won't detect `data[16]` as an occurrence, but will find `element_{1,2,3}`. If we combine these results with the Highlighted dump feature, we will be able to determine easily we have a 16 bytes fixed string buffer.

But PmatchS won't find `element_3` as an occurrence since `data` is variable length:

```
typedef struct test {
    uint32_t    element_1;
    uint16_t    element_2;
    char        *data;
    uint32_t    element_3;
};
```

4.2 Multiple files parsing

Assuming we want to compare several files such as `ndrdump`³ files⁴ to see if we have fixed data, PmatchS can't perform this operation since it doesn't support multiple files searching. Instead you can make a concatenation of multiple files into a single one.

³`ndrdump` is part of the Samba4 Test Suite

⁴`ndrdump` files as `pcap` files, like the stub data you export from Ethereal

5 Planned features

5.1 Algorithm plugin system

The algorithm is currently hard coded. We will migrate algorithms into external plugins, so the software can be extended easily.

5.2 XML element structure parsing

Developers analysing database will be able to define fixed or dynamic data structure and check whether their mapping is correct or not.

5.3 Developer API

The algorithm plugin system and the XML parsing will be supplied with an API allowing developers to develop new plugins and add new xml parsing file to the project database.

5.4 Highlighted dump enhancements

A caption detailing the color/relevance association needs to be added.

5.5 Configuration file

We will add a configuration file for PmatchS

6 Bug Report and features

Please do not hesitate to contact us if you think you have found a bug or limitation.

6.1 Contact

You can contact us using:

- OpenReverse discussion board on Sourceforge:
- openchange-devel mailing list: devel@openchange.org
- mail: pmatches@openchange.org