恶意代码分析与防治技术

第3章 Yara检测引擎

王志
zwang@nankai.edu.cn

南开大学 网络空间安全学院

2023-2024学年

本章知识点

- Yara引擎
- Yara引擎安装
- Yara规则
- Yara字符串
    - 难点：正则表达式
- Yara条件表达式

1. Yara引擎

主观题 10分

设置

第2章恶意代码基本静态分析技术中介绍了哪些静态分析方法？这些静态分析方法能提取出哪些恶意代码特征?

作答

主观题 10分

设置

构造一个杀毒软件，除了特征还需要什么?

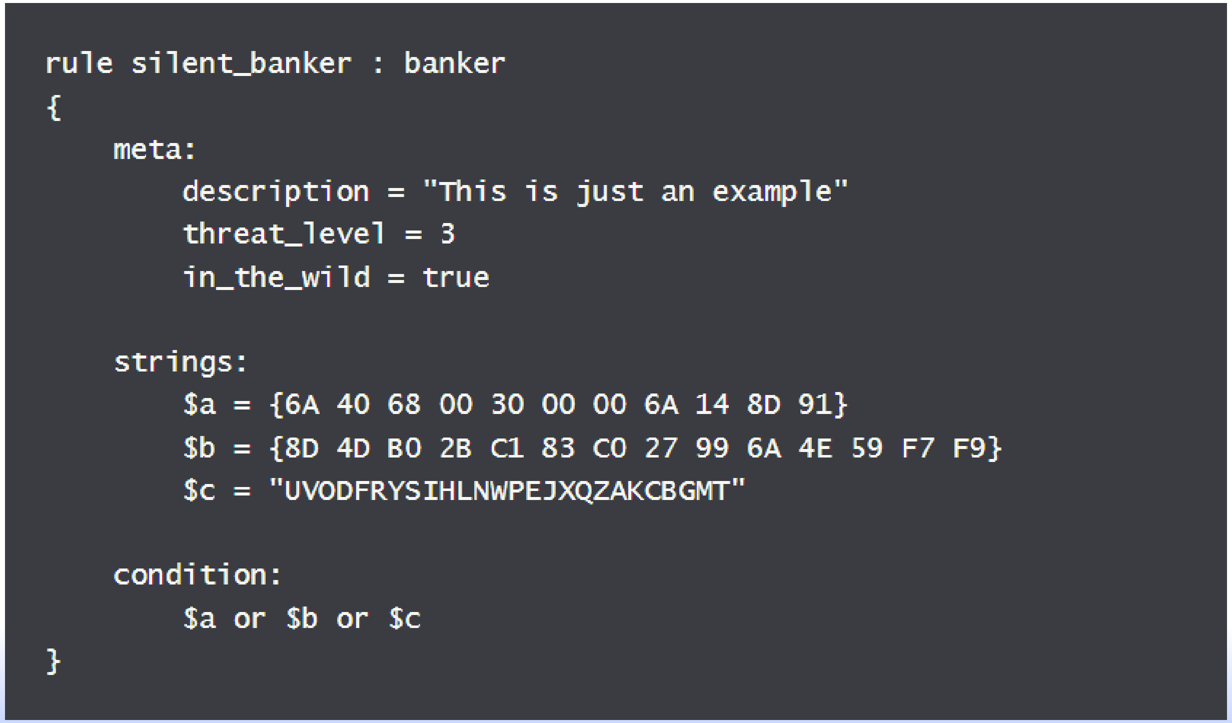
作答

Yara引擎

- Yara 是VirusTotal发布的一个开源恶意代码查杀引擎
  - 识别恶意代码
  - 分类恶意代码
  - https://virustotal.github.io/yara/
- Yara 引擎是跨平台的，可在 Windows、Linux 和 Mac OS X 上运行。

Yara引擎

- Yara 本身不提供杀毒功能
  - 没有特征库
  - 需要编写Yara规则，以此来识别和分类恶意软件或者程序。



```
rule silent_banker : banker
{
    meta:
        description = "This is just an example"
        threat_level = 3
        in_the_wild = true

    strings:
        $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
        $b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
        $c = "UVODFRYSIHLNWPEJXQZAKCBGMT"

    condition:
        $a or $b or $c
}
```

Yara规则

- Yara规则是由**一系列**字符串和**一个**布尔型表达式构成
- 支持与或非等多种条件

```
rule silent_banker : banker
{
    meta:
        description = ″This is just an example″

    strings:
        $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
        $b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
        $c = ″UVODFRYSIHLNWPEJXQZAKCBGMT″

    condition:
        $a or $b or $c
}
```

多选题　10分

设置

以下对Yara引擎的描述哪些是正确的？

A　可以跨平台

B　可以用来识别和分类恶意代码

C　Yara引擎本身不提供杀毒功能

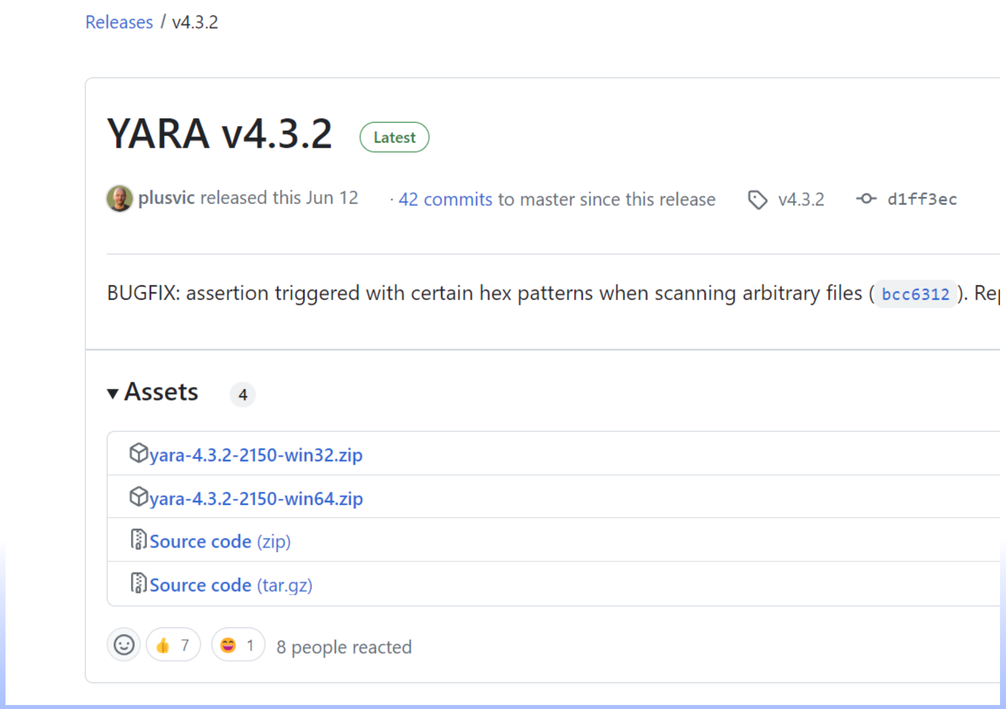
D　Yara规则由1组字符串和1个布尔表达式构成

提交

2. Yara引擎的安装

Yara引擎

- Yara引擎的github地址：
  - https://github.com/VirusTotal/yara/releases



Releases / v4.3.2

YARA v4.3.2 Latest

plusvic released this Jun 12 · 42 commits to master since this release v4.3.2 d1ff3ec

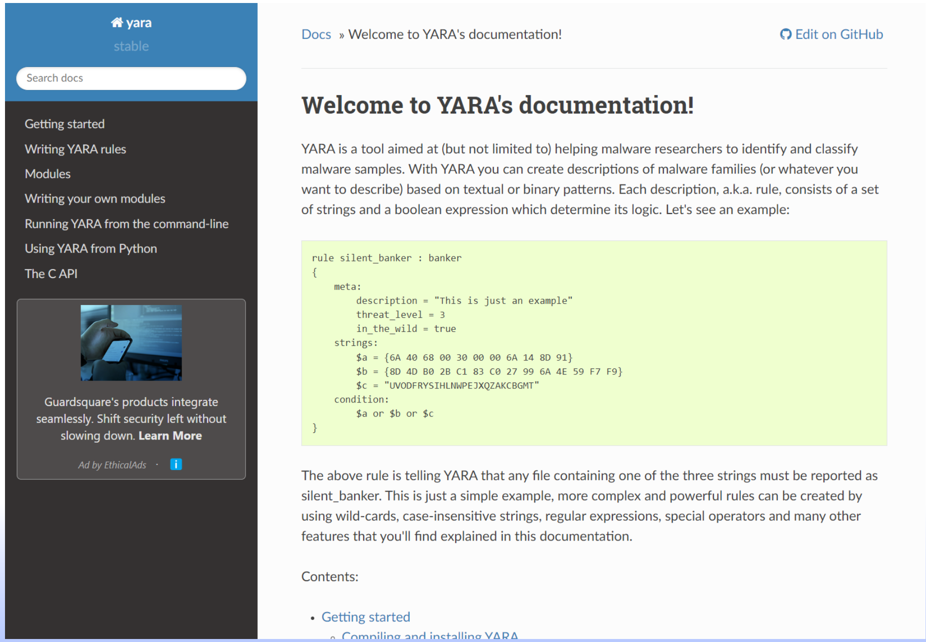
BUGFIX: assertion triggered with certain hex patterns when scanning arbitrary files (bcc6312). Re

▾ Assets 4

- yara-4.3.2-2150-win32.zip
- yara-4.3.2-2150-win64.zip
- Source code (zip)
- Source code (tar.gz)

Yara引擎

- Yara引擎文档说明：
  - https://yara.readthedocs.io/en/stable



Welcome to YARA's documentation!

YARA is a tool aimed at (but not limited to) helping malware researchers to identify and classify malware samples. With YARA you can create descriptions of malware families (or whatever you want to describe) based on textual or binary patterns. Each description, a.k.a. rule, consists of a set of strings and a boolean expression which determine its logic. Let's see an example:

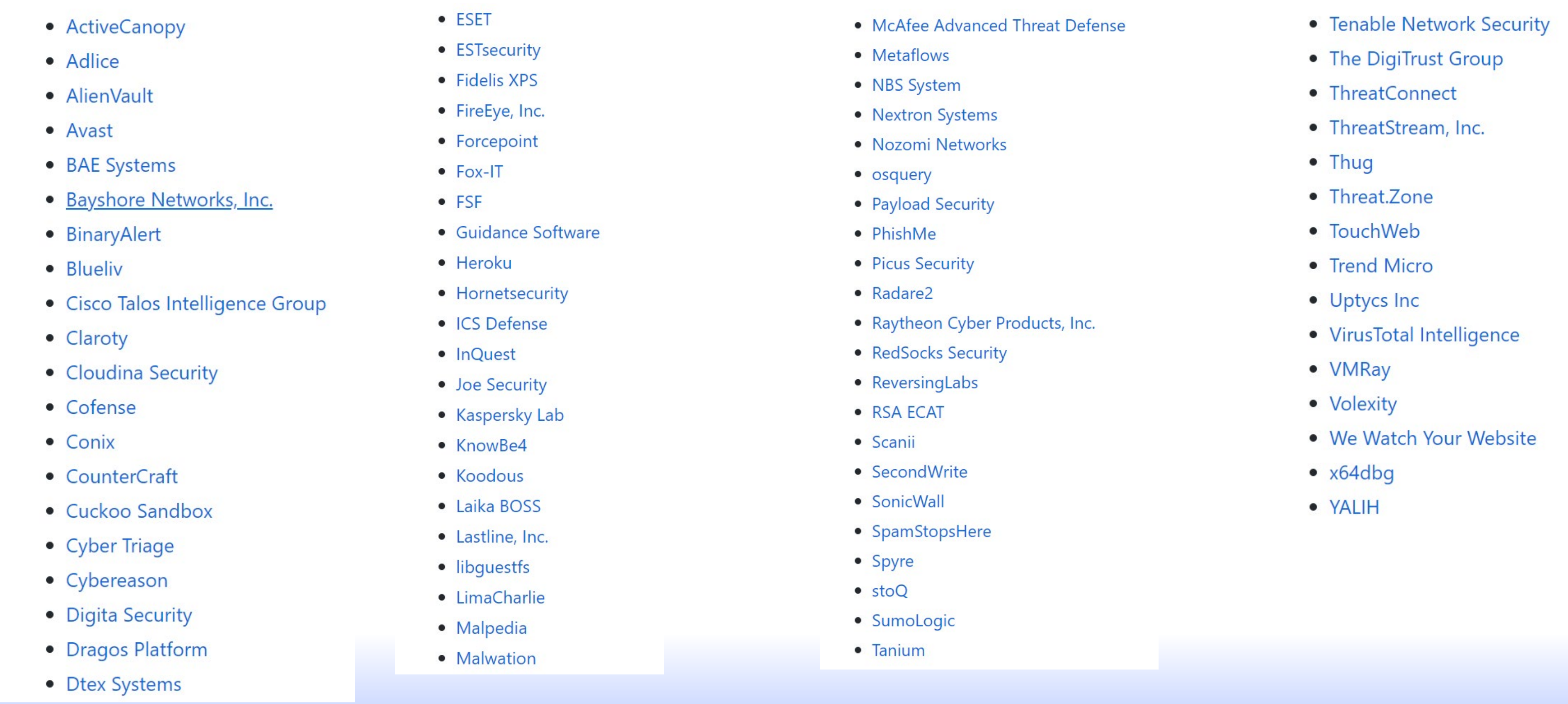
```
rule silent_banker : banker
{
    meta:
        description = "This is just an example"
        threat_level = 3
        in_the_wild = true
    strings:
        $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
        $b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
        $c = "UVODFRYSIHLNWPEJXQZAKCBGMT"
    condition:
        $a or $b or $c
}
```

The above rule is telling YARA that any file containing one of the three strings must be reported as silent_banker. This is just a simple example, more complex and powerful rules can be created by using wild-cards, case-insensitive strings, regular expressions, special operators and many other features that you'll find explained in this documentation.

Contents:

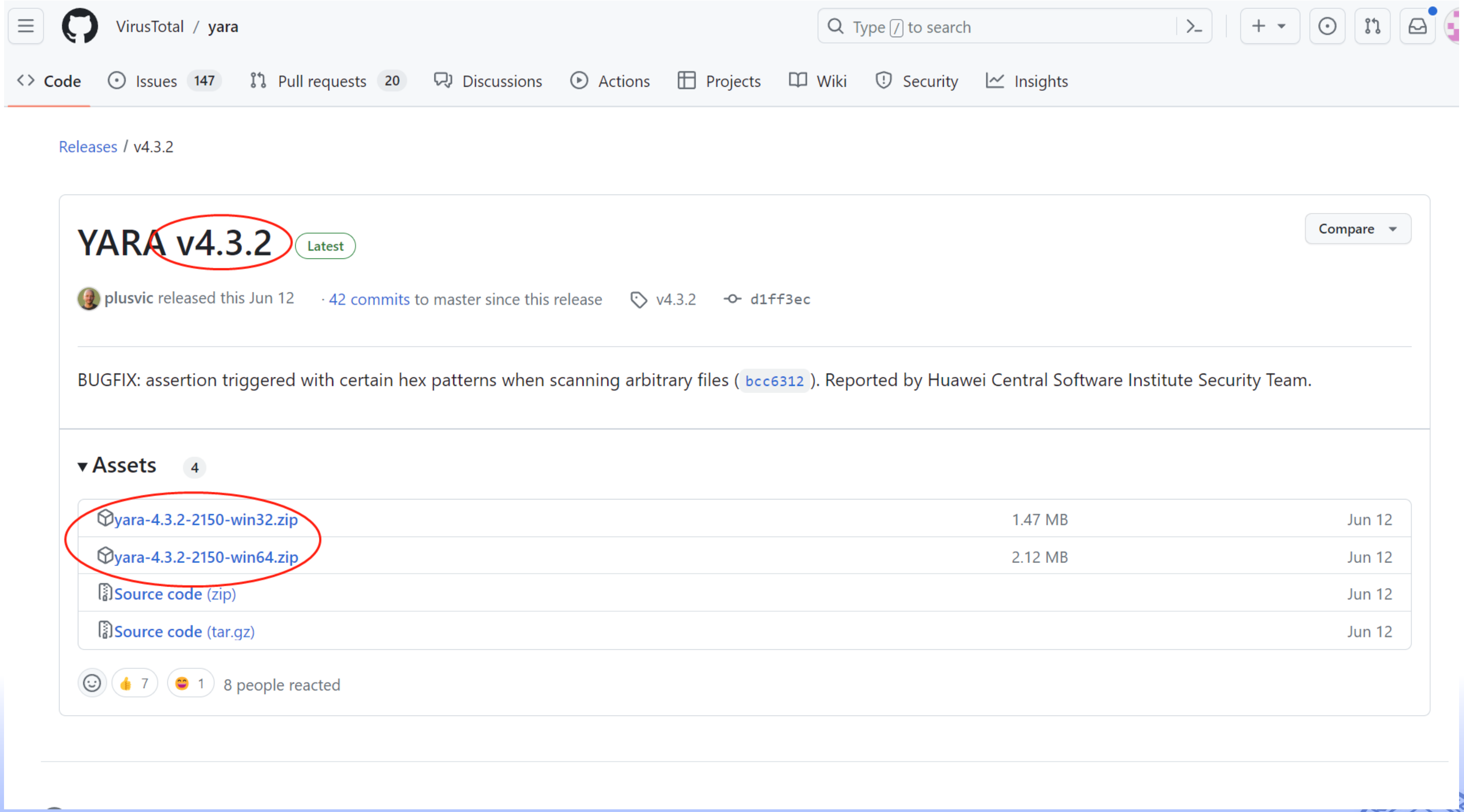
- Getting started
  - Compiling and installing YARA

Who is using Yara?



- ActiveCanopy
- Adlice
- AlienVault
- Avast
- BAE Systems
- Bayshore Networks, Inc.
- BinaryAlert
- Blueliv
- Cisco Talos Intelligence Group
- Claroty
- Cloudina Security
- Cofense
- Conix
- CounterCraft
- Cuckoo Sandbox
- Cyber Triage
- Cybereason
- Digita Security
- Dragos Platform
- Dtex Systems
- ESET
- ESTsecurity
- Fidelis XPS
- FireEye, Inc.
- Forcepoint
- Fox-IT
- FSF
- Guidance Software
- Heroku
- Hornetsecurity
- ICS Defense
- InQuest
- Joe Security
- Kaspersky Lab
- KnowBe4
- Koodous
- Laika BOSS
- Lastline, Inc.
- libguestfs
- LimaCharlie
- Malpedia
- Malwation
- McAfee Advanced Threat Defense
- Metaflows
- NBS System
- Nextron Systems
- Nozomi Networks
- osquery
- Payload Security
- PhishMe
- Picus Security
- Radare2
- Raytheon Cyber Products, Inc.
- RedSocks Security
- ReversingLabs
- RSA ECAT
- Scanii
- SecondWrite
- SonicWall
- SpamStopsHere
- Spyre
- stoQ
- SumoLogic
- Tanium
- Tenable Network Security
- The DigiTrust Group
- ThreatConnect
- ThreatStream, Inc.
- Thug
- Threat.Zone
- TouchWeb
- Trend Micro
- Uptycs Inc
- VirusTotal Intelligence
- VMRay
- Volexity
- We Watch Your Website
- x64dbg
- YALIH

Windows安装Yara引擎



Releases / v4.3.2

YARA v4.3.2 Latest

plusvic released this Jun 12 · 42 commits to master since this release v4.3.2 d1ff3ec

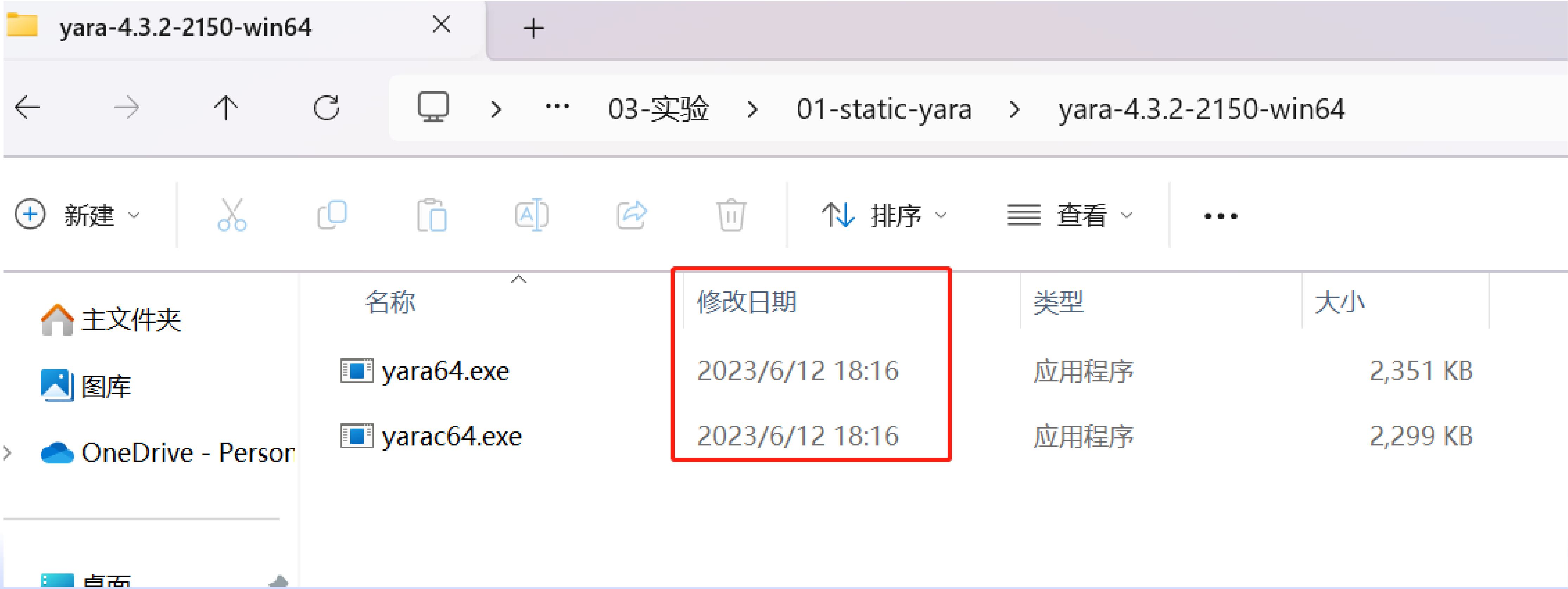
BUGFIX: assertion triggered with certain hex patterns when scanning arbitrary files (`bcc6312`). Reported by Huawei Central Software Institute Security Team.

▾ Assets 4

yara-4.3.2-2150-win32.zip	1.47 MB	Jun 12
yara-4.3.2-2150-win64.zip	2.12 MB	Jun 12
Source code (zip)		Jun 12
Source code (tar.gz)		Jun 12

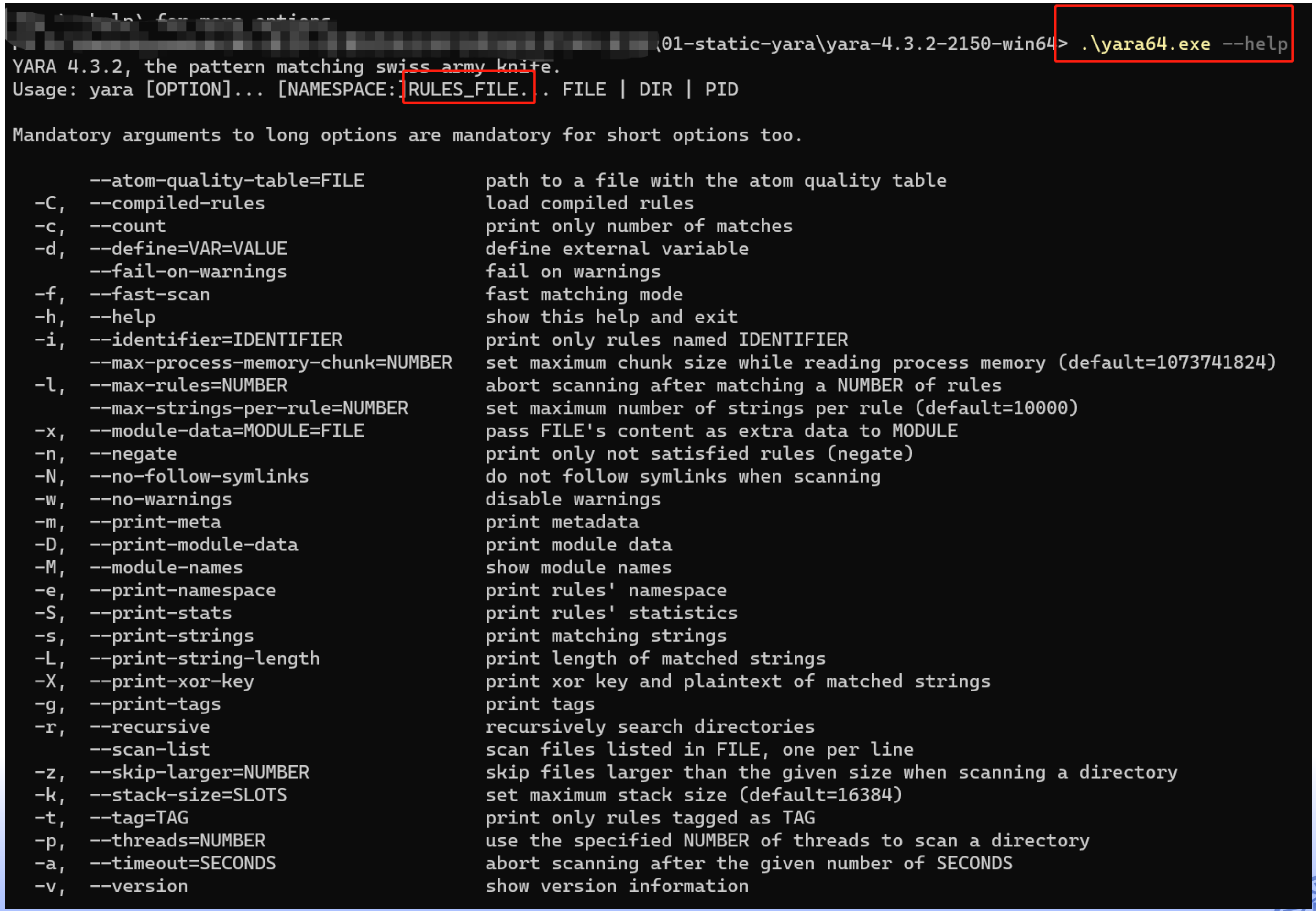
👍 7 😄 1 8 people reacted

Yara 引擎



名称	修改日期	类型	大小
yara64.exe	2023/6/12 18:16	应用程序	2,351 KB
yarac64.exe	2023/6/12 18:16	应用程序	2,299 KB

Windows安装Yara引擎

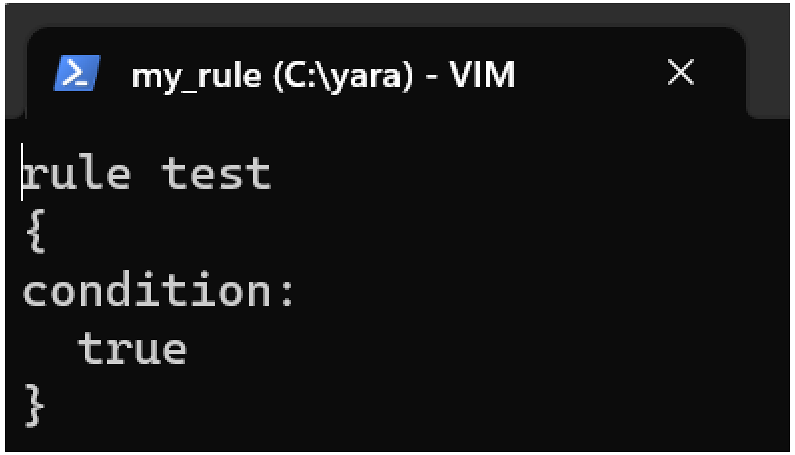


```
[redacted]lp) for more options.
[redacted]\01-static-yara\yara-4.3.2-2150-win64> .\yara64.exe --help
YARA 4.3.2, the pattern matching swiss army knife.
Usage: yara [OPTION]... [NAMESPACE:]RULES_FILE... FILE | DIR | PID

Mandatory arguments to long options are mandatory for short options too.

       --atom-quality-table=FILE           path to a file with the atom quality table
  -C,  --compiled-rules                    load compiled rules
  -c,  --count                             print only number of matches
  -d,  --define=VAR=VALUE                  define external variable
       --fail-on-warnings                  fail on warnings
  -f,  --fast-scan                         fast matching mode
  -h,  --help                              show this help and exit
  -i,  --identifier=IDENTIFIER             print only rules named IDENTIFIER
       --max-process-memory-chunk=NUMBER   set maximum chunk size while reading process memory (default=1073741824)
  -l,  --max-rules=NUMBER                  abort scanning after matching a NUMBER of rules
       --max-strings-per-rule=NUMBER       set maximum number of strings per rule (default=10000)
  -x,  --module-data=MODULE=FILE           pass FILE's content as extra data to MODULE
  -n,  --negate                            print only not satisfied rules (negate)
  -N,  --no-follow-symlinks                do not follow symlinks when scanning
  -w,  --no-warnings                       disable warnings
  -m,  --print-meta                        print metadata
  -D,  --print-module-data                 print module data
  -M,  --module-names                      show module names
  -e,  --print-namespace                   print rules' namespace
  -S,  --print-stats                       print rules' statistics
  -s,  --print-strings                     print matching strings
  -L,  --print-string-length               print length of matched strings
  -X,  --print-xor-key                     print xor key and plaintext of matched strings
  -g,  --print-tags                        print tags
  -r,  --recursive                         recursively search directories
       --scan-list                         scan files listed in FILE, one per line
  -z,  --skip-larger=NUMBER                skip files larger than the given size when scanning a directory
  -k,  --stack-size=SLOTS                  set maximum stack size (default=16384)
  -t,  --tag=TAG                           print only rules tagged as TAG
  -p,  --threads=NUMBER                    use the specified NUMBER of threads to scan a directory
  -a,  --timeout=SECONDS                   abort scanning after the given number of SECONDS
  -v,  --version                           show version information
```

安装yara-python

- 直接编写python程序来调用Yara引擎
- pip自动安装
  - pip install yara-python

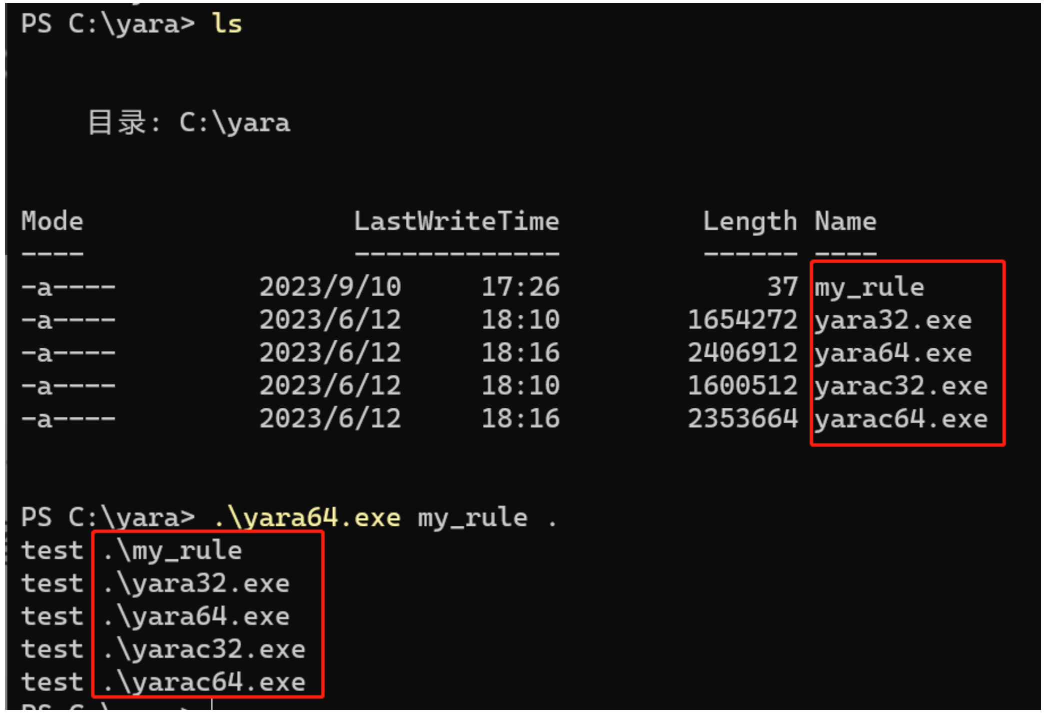
运行Yara引擎



my_rule (C:\yara) - VIM

```
rule test
{
condition:
   true
}
```

编写规则test，判断条件写true，
所有的文件都会被匹配



```
PS C:\yara> ls


    目录: C:\yara


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          2023/9/10     17:26             37 my_rule
-a----          2023/6/12     18:10        1654272 yara32.exe
-a----          2023/6/12     18:16        2406912 yara64.exe
-a----          2023/6/12     18:10        1600512 yarac32.exe
-a----          2023/6/12     18:16        2353664 yarac64.exe


PS C:\yara> .\yara64.exe my_rule .
test .\my_rule
test .\yara32.exe
test .\yara64.exe
test .\yarac32.exe
test .\yarac64.exe
```

当前文件夹下的所有文件都被
test规则匹配

3. Yara规则

Yara规则

规则开始的关键字

```
rule dummy
{
    condition:
        false
}
```

规则标识符Identifier，第一个字符不能是数字，长度不超过128字符，区分大小写

Yara规则类似于C语言，每个规则都以关键字“rule”开头

Yara规则中的关键字

all	and	any	ascii	at	base64	base64wide	condition
contains	entrypoint	false	filesize	for	fullword	global	import
in	include	int16	int16be	int32	int32be	int8	int8be
matches	meta	nocase	not	of	or	private	rule
strings	them	true	uint16	uint16be	uint32	uint32be	uint8
uint8be	wide	xor					

```
rule silent_banker : banker  tag字段
{        规则名
meta:                          描述信息
        description = "This is just an example"
strings:  规则字段
        $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
        $b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
        $c = "UVODFRYSIHLNWPEJXQZAKCBGMT"
condition:
        $a or $b or $c    条件判断字段
}
```

注释

- 可以像编写C语言一样，在Yara规则中添加注释：
  - // 单行注释
  - /* 多行注释 */

多选题 10分

设置

下面哪一个是无效的规则名称?

[✓ A] 00_banker

B Trojan_0x234

C My_first_rule

D silent_banker:banker

提交

4. Yara字符串

字符串

- Yara中有三种类型的字符串：
  - **十六进制串**：定义原始字节序列

    $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
  - **文本字符串**：定义可读文本的部分

    "UVODFRYSIHLNWPEJXQZAKCBGMT"
  - **正则表达式**：定义可读文本的部分

通配符

```
rule WildcardExample
{
strings: //使用‘?’作为通配符
    $hex_string = { 00 11 ?? 33 4? 55 }
condition:
    $hex_string
}
```

跳转

```
rule JumpExample
{
strings:
//使用‘[]’作为跳转，与任何长度为0-2字节的内容匹配
        $hex_string1 = { 00 11 [2] 44 55 }
        $hex_string2 = { 00 11 [0-2] 44 55 }
        //该写法与string1作用完全相同
        $hex_string3 = { 00 11 ?? ?? 44 55 }
condition:
        $hex_string1 and $hex_string2
}
```

正则表达式

```
rule AlternativesExample1
{
strings:
      $hex_string = { 00 11 ( 22 | 33 44 ) 55 }
/* 匹配 00 11  22  55 或者 00 11  33 44  55 */
condition:
      $hex_string
}
```

正则表达式

```
rule AlternativesExample2
{
strings:
        $hex_string = { 00 11 ( 33 44 | 55 | 66 ?? 88 ) 99 }
condition:
        $hex_string
}
```

多选题　10分

设置

$hex_string = { 00 11 (33 44 | 55 | 66 ?? 88) 99 }

该yara规则可以匹配以下哪几个十六进制串

A　00 11 33 44

[✓ B]　00 11 55 99

C　00 11 66 77 88

[✓ D]　00 11 66 56 88 99

提交

主观题 10分

设置

文本字符串的匹配要考虑哪些问题？

作答

修饰符

● nocase：	不区分大小写
● wide：	匹配2字节的宽字符
● ascii：	匹配1字节的ascii字符
● xor：	匹配异或后的字符串
● base64：	匹配Base64编码的字符串
● fullword：	匹配完整单词
● private：	定义私有字符串

文本字符串-转义符

\″	双引号
\\	反斜杠
\t	制表符
\n	换行符
\xdd	十六进制的任何字节

文本字符串

- 不区分大小写

```
$text_string = ″foobar″ nocase
```

- 匹配宽字符串

```
$wide_string = ″Borland″ wide
```

- 同时匹配2种类型的字符串

```
$wide_and_ascii_string = ″Borland″ wide ascii
```

文本字符串

- 匹配所有可能的异或后字符串

```
$xor_string = ″This program cannot″ xor
```

- 匹配所有可能的异或后wide和ascii字符串

```
$xor_string = ″This program cannot″ xor wide ascii
```

- 限定异或范围

```
$xor_string = ″This program cannot″ xor(0x01-0xff)
```

文本字符串

- 匹配base64编码的字符串
  - $a = "This program cannot" base64
    - VGhpcyBwcm9ncmFtIGNhbm5vd
    - RoaXMgcHJvZ3JhbSBjYW5ub3
    - UaGlzIHByb2dyYW0gY2Fubm90
  - $a = "This program cannot"

    base64("!@#$%^&*(){}[].,|ABCDEFGHIJ\x09LMNOPQRSTUVWXYZabcdefghijklmnopqrstu")
    - 自定义Base64编码的字母表

文本字符串

- 全词匹配

```
$wide_string = ″domain″ fullword
```

匹配:www.domain.com， www.my-domain.com

不匹配:www.mydomain.com

- 私有字符串：正常匹配规则，不会在输出中显示

```
$text_string = ″foobar″ private
```

多选题 10分

设置

$wide_string = “nankai” fullword，以下哪些字符串会匹配$wide_string

A www.nankai.edu.cn

B ilovenankai

C i-nankai

D nankai university

提交

正则表达式-元字符（metacharacters）

\	Quote the next metacharacter
^	Match the beginning of the file
$	Match the end of the file
|	Alternation
()	Grouping
[]	Bracketed character class

数量匹配（quantifiers）

*	Match 0 or more times
+	Match 1 or more times
?	Match 0 or 1 times
{n}	Match exactly n times
{n,}	Match at least n times
{,m}	Match at most m times
{n,m}	Match n to m times

字符类型定义

\w	Match a *word* character (alphanumeric plus “_”)
\W	Match a *non-word* character
\s	Match a whitespace character
\S	Match a non-whitespace character
\d	Match a decimal digit character
\D	Match a non-digit character

5. Yara条件表达式

主观题 10分

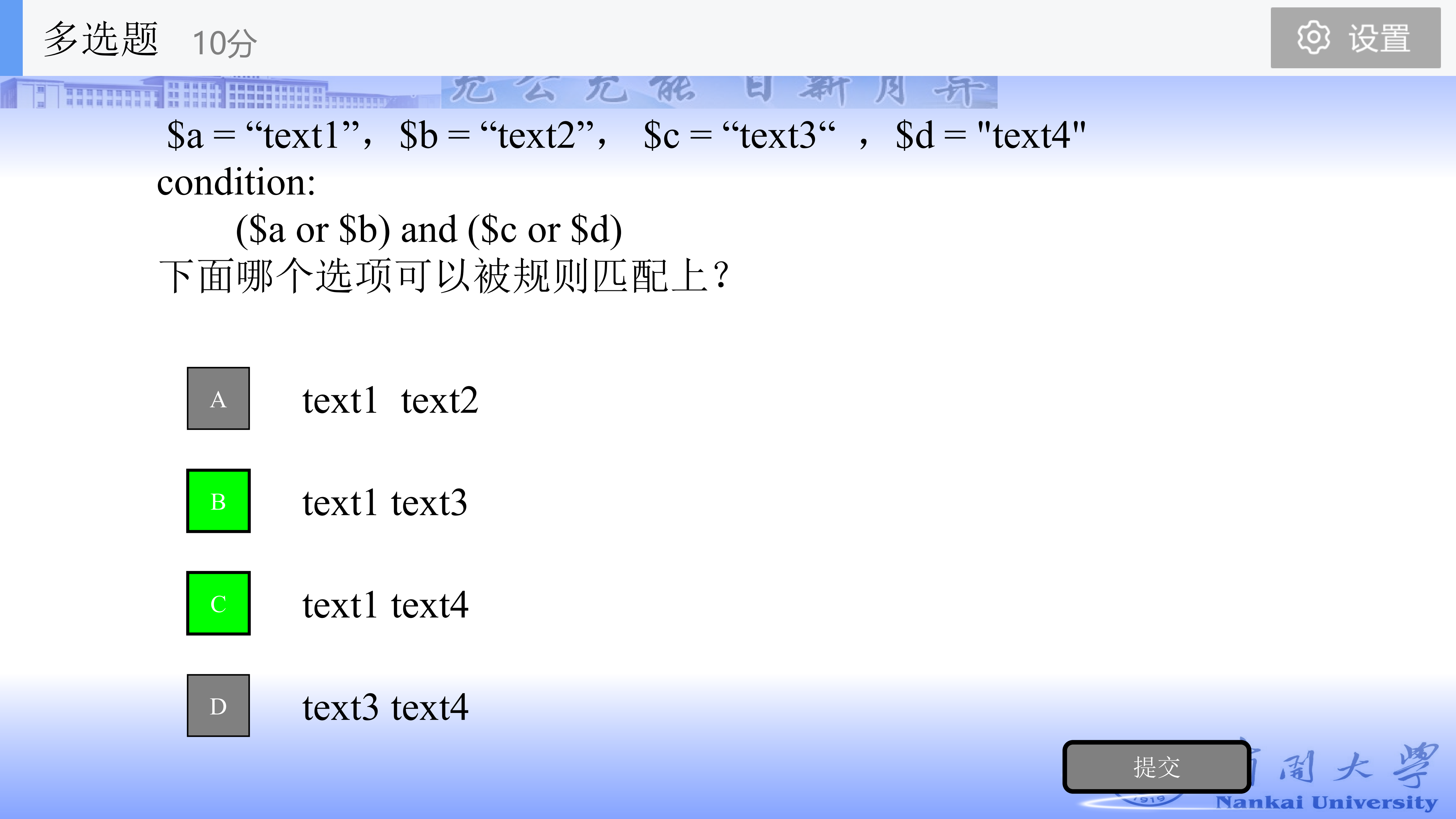
设置

在条件表达式中，有哪些可能的特征字符串组合方式?

作答

条件表达式

- 布尔表达式
- 布尔操作符：and、or、not
- 关系操作符：>=、<=、<、>==、！＝
- 位操作符：&、|、<<、>>、~、^



多选题 10分

$a = “text1”， $b = “text2”， $c = “text3“ ， $d = "text4"

condition:

  ($a or $b) and ($c or $d)

下面哪个选项可以被规则匹配上？

A text1 text2

[✓ B] text1 text3

[✓ C] text1 text4

D text3 text4

# Counting strings

```
strings:
    $a = ″dummy1″
    $b = ″dummy2″
condition:
    //a字符串出现6次，b字符串大于10次
    #a == 6 and #b > 10
```

@ 获取字符串出现位置

- 可以使用@a[*i*]，获取字符串$a在文件或内存中，第*i*次出现的偏移或虚拟地址。
- 索引从1开始，不是从0开始。如果*i*大于字符串出现的次数，结果为NaN(not a number 非数值)。

！ 获取字符串长度

- 可以使用!a[*i*]，获取字符串$a在文件或内存中，第*i*次出现时的字符串长度。
- 索引同@一样都是从1开始，不是从0开始。
- !a 是 !a[1]的简写。

at 指定字符串匹配的位置

- at 匹配字符串在文件或内存中的偏移

```
strings:
      $a = ″dummy1″
      $b = ″dummy2″
condition: //a和b字符串出现在文件或内存的100和200偏移处
      $a at 100 and $b at 200
```

in 指定字符串匹配的范围

- **in** 在文件或内存的某个地址范围内匹配字符串

```
strings:
    $a = ″dummy1″
    $b = ″dummy2″
condition:
    $a in (0..100) and $b in (100..filesize)
```

filesize 文件大小匹配

- filesize匹配文件大小

```
condition:
    //filesize只在文件时才有用，对进程无效
    //KB MB后缀只能与十进制大小一起使用
    filesize > 200KB
```

entrypoint 入口点匹配

- 匹配PE或ELF文件入口点(高版本使用PE模块的pe.entry_point代替)

```
strings:
    $a = { E8 00 00 00 00 }
condition:
    $a at entrypoint
```

entrypoint

```
strings:
    $a = { 9C 50 66 A1 ?? ?? ?? 00 66 A9 ?? ?? 58
0F 85 }
condition:
    $a in (entrypoint..entrypoint + 10)
```

读文件或内存数据

- **intxxx**读取小端有符号整数
- int8(<offset or virtual address>)
- int16(<offset or virtual address>)
- int32(<offset or virtual address>)

读文件或内存数据

- **uintxxx**读取小端无符号整数
- uint8(<offset or virtual address>)
- uint16(<offset or virtual address>)
- uint32(<offset or virtual address>)

读文件或内存数据

- **intxxxbe**读取大端有符号整数
- int8be(<offset or virtual address>)
- int16be(<offset or virtual address>)
- int32be(<offset or virtual address>)

读内存或文件数据

- **uintxxxbe**读取大端无符号整数
- uint8be(<offset or virtual address>)
- uint16be(<offset or virtual address>)
- uint32be(<offset or virtual address>)

主观题 10分

设置

编写判断文件是否是PE文件的Yara规则。

IMAGE_DOS_SIGNATURE = 0x5A4D

e_lfanew 的偏移地址是0x3C

IMAGE_NT_SIGNATURE = 0x00004550

正常使用主观题需2.0以上版本雨课堂

作答

IsPE

```
rule IsPE
{
condition:
        //判断是否PE文件
        uint16(0) == 0x5A4D and // “MZ” 头
        uint32(uint32(0x3C)) == 0x00004550 //  “PE” 头
}
```

# of 匹配部分字符串

- 匹配多个字符串中的某几个

```
strings:
    $a = ″dummy1″
    $b = ″dummy2″
    $c = ″dummy3″
 condition: //3个字符串只需匹配任意2个
    2 of ($a, $b, $c)
```

for 多字符串匹配

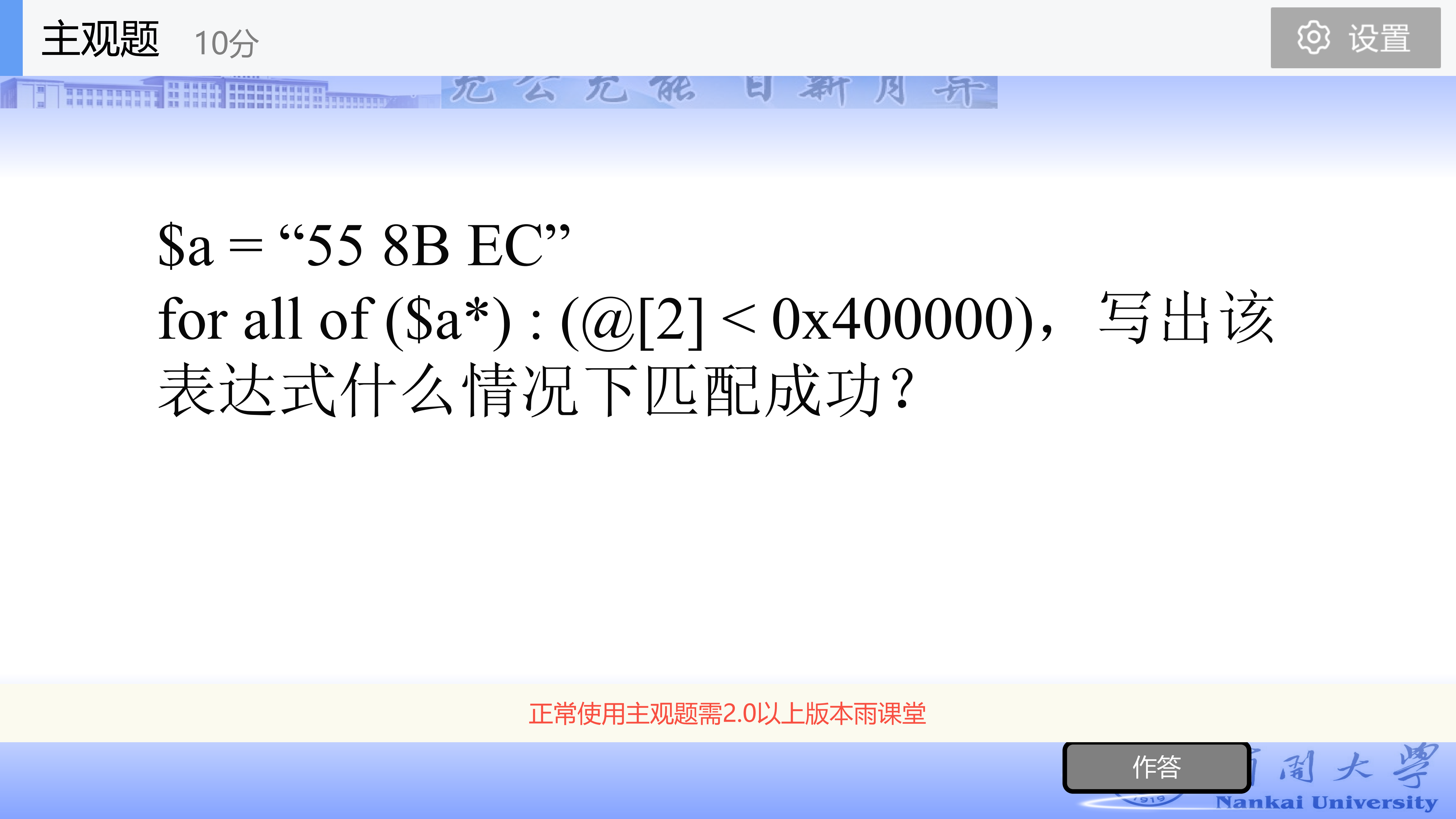
- for AAA of BBB : (CCC)
- 在BBB字符串集合中，至少有AAA个字符串，满足了CCC的条件表达式，才算匹配成功。

  for 1 of （$a, $b, $c） : (# > 3)

  //至少1个字符串在文件或内存中出现的次数大于3

any、all、them 多字符串匹配

- 在条件表达式中，可以使用$依次代替字符串集合中的每一个字符串，#表示字符串的出现次数
- for 1 of （$a,$b,$c） : （ $ at entrypoint ）
- for any of （$a,$b,$c） : （ $ at entrypoint ）
- for all of them : （ # > 3 ）



主观题 10分

$a = “55 8B EC”

for all of ($a*) : (@[2] < 0x400000)， 写出该表达式什么情况下匹配成功？

正常使用主观题需2.0以上版本雨课堂

for-in 多字符串匹配

- for AAA BBB in (CCC) : (DDD)
- 作用与for of类似，增加了下标变量与下标范围

for all i in (1,2,3) : (@a[i] + 10 == @b[i])

主观题 10分

设置

$a每次在文件或内存中出现位置，都必须小于100。
（使用for-in表达方式来描述）

正常使用主观题需2.0以上版本雨课堂

作答

引用其它规则

```
strings:
        $a = "dummy2"

condition:
        $a and IsPE
```

全局规则

- 全局规则（global rule）可以在匹配其他规则前优先筛选

```
global rule SizeLimit
{
condition:
    filesize < 2MB
}
```

比如在匹配目标文件之前需要先筛选出小于2MB的文件，再匹配其它规则。

私有规则

- 私有规则（private rule）避免规则匹配结果的混乱，YARA不会输出任何匹配到的私有规则信息。

```
private rule PrivateRuleExample
{
...
}
```

导入模块

- import导入模块，可以使用第三方模块导出的变量或函数

```
import ″pe″
import ″cuckoo″
pe.entry_point == 0x1000
cuckoo.http_request(/someregexp/)
```

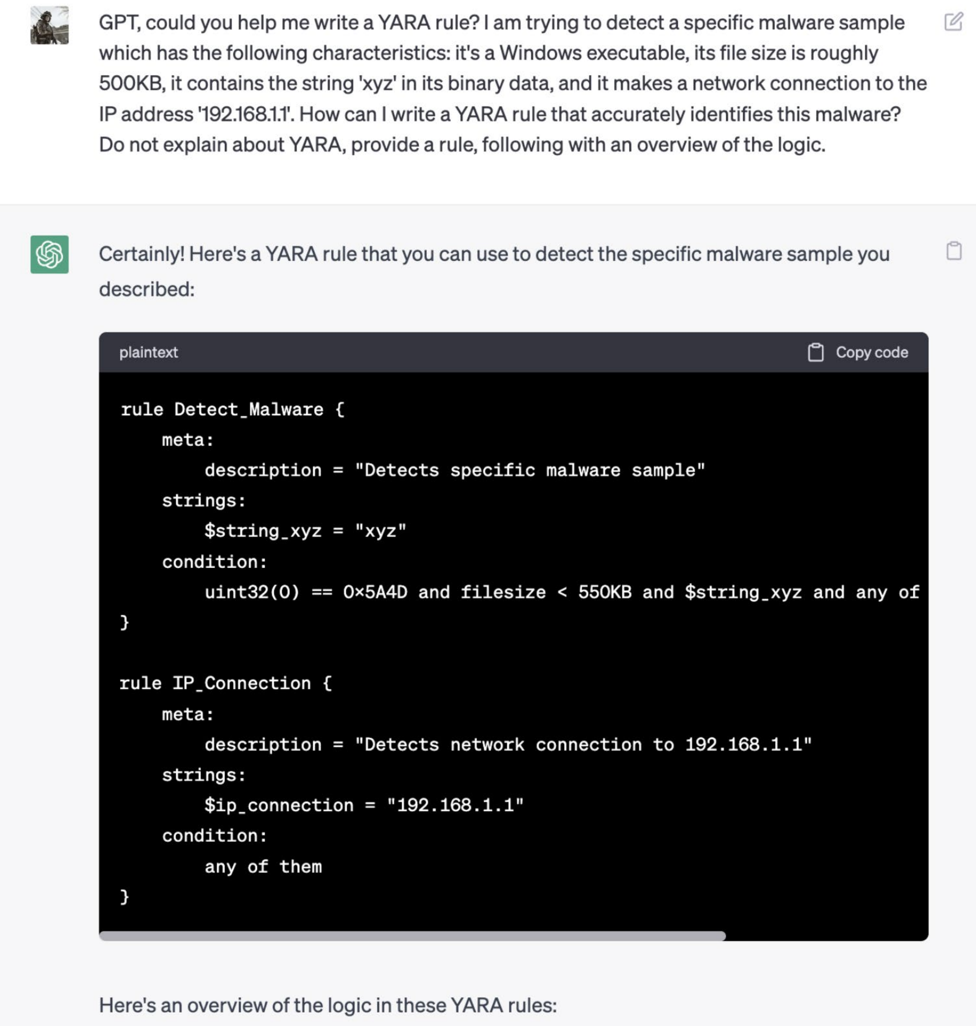
外部变量

- 外部变量允许使用YARA -d命令时指定一个自定义数据
- 该数据可以是整数、字符串、布尔变量

文件包含

- include包含其它规则文件的内容到当前文件中
- 相对路径
  - include ″./includes/other.yar″
  - include ″../includes/other.yar″
- 全路径
  - include ″/home/plusvic/yara/includes/other.yar″

ChatGPT

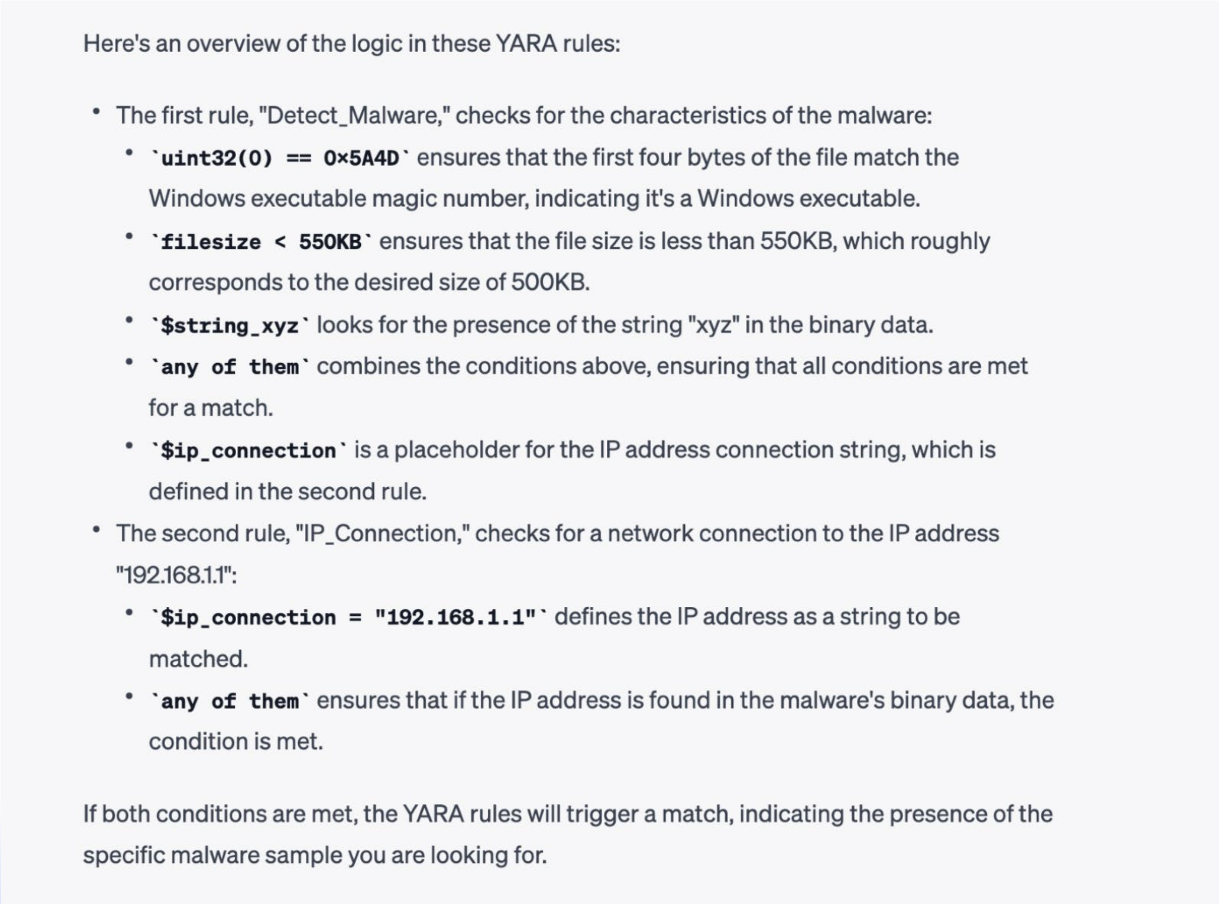


GPT, could you help me write a YARA rule? I am trying to detect a specific malware sample which has the following characteristics: it's a Windows executable, its file size is roughly 500KB, it contains the string 'xyz' in its binary data, and it makes a network connection to the IP address '192.168.1.1'. How can I write a YARA rule that accurately identifies this malware? Do not explain about YARA, provide a rule, following with an overview of the logic.

Certainly! Here's a YARA rule that you can use to detect the specific malware sample you described:

```
rule Detect_Malware {
    meta:
        description = "Detects specific malware sample"
    strings:
        $string_xyz = "xyz"
    condition:
        uint32(0) == 0x5A4D and filesize < 550KB and $string_xyz and any of
}

rule IP_Connection {
    meta:
        description = "Detects network connection to 192.168.1.1"
    strings:
        $ip_connection = "192.168.1.1"
    condition:
        any of them
}
```

Here's an overview of the logic in these YARA rules:



Here's an overview of the logic in these YARA rules:

- The first rule, "Detect_Malware," checks for the characteristics of the malware:
  - `uint32(0) == 0x5A4D` ensures that the first four bytes of the file match the Windows executable magic number, indicating it's a Windows executable.
  - `filesize < 550KB` ensures that the file size is less than 550KB, which roughly corresponds to the desired size of 500KB.
  - `$string_xyz` looks for the presence of the string "xyz" in the binary data.
  - `any of them` combines the conditions above, ensuring that all conditions are met for a match.
  - `$ip_connection` is a placeholder for the IP address connection string, which is defined in the second rule.
- The second rule, "IP_Connection," checks for a network connection to the IP address "192.168.1.1":
  - `$ip_connection = "192.168.1.1"` defines the IP address as a string to be matched.
  - `any of them` ensures that if the IP address is found in the malware's binary data, the condition is met.

If both conditions are met, the YARA rules will trigger a match, indicating the presence of the specific malware sample you are looking for.

实验课

- 对Lab1的样本编写Yara检测规则，并进行测试。
- 在雨课堂上提交实验报告。
- 加分项：讨论如何编写更加快速的yara规则？

南開大學
NANKAI UNIVERSITY,P.R.CHINA 1919
允公允能 日新月异

恶意代码分析与防治技术

第3章 Yara检测引擎

王志
zwang@nankai.edu.cn

南开大学 网络空间安全学院

2023-2024学年