# 第 3 章 CyberChef 的数据处理

CHAPTER 3

CyberChef 提供了用户友好界面，其拖放式操作界面让用户无须编程技能即可构建复杂的数据处理流程。同时，它具有丰富的功能模块，涵盖从编码、解码、加密、解密、数据转换、压缩、解压等多种功能。CyberChef 的实时反馈能够将结果实时显示，便于用户即时查看和调整操作。作为一个开源工具，它具有灵活性和社区支持，可以满足不同的数据处理需求。本章将介绍关于对比、提取、格式化数据的常规操作，以及通过正则表达式进行匹配操作的数据处理方法。

## 3.1 数据的常规操作

数据是信息的原材料。通过处理和分析数据，能够提取出有意义的信息，从而使数据的潜在价值得以显现。常规的数据处理方法有对比、提取、格式化等。对比数据有利于发现差异，从而验证信息的准确性，进行质量控制或找到数据中的异常。提取数据是指从原始数据中提取关键信息，是确保有效分析的基础，能帮助用户找到和利用有价值的数据片段。格式化数据可以使其结构化、标准化，便于进一步分析、处理或可视化，从而提高数据处理的效率和准确性。

### 3.1.1 对比数据

对比数据在数据处理中是指通过对比两个或多个数据集、数据记录或数据版本，识别它们之间的异同。通过对比数据能够发现数据中的差异、变化或异常，验证数据的准确性。有效的数据对比能够提升数据分析的精确度，帮助识别问题和优化决策过程。笔者认为对比数据的意义在于能够找到其中的不同点，从而找到可用信息。

CyberChef 提供的 Diff 模块用于比较两个数据片段或字符串，找出它们之间的差异。这对于数据比较和分析非常有用，尤其是在检查文件的版本差异、调试程序输出、验证数据变更等场景中。总之，使用 CyberChef 的 Diff 模块，可以有效地分析和处理数据差异，提高数据处理和分析的效率。

Diff 模块能够通过 Sample delimiter 参数来设置两个不同数据的分隔符，默认为\n\n，其中，\n\n 表示两个换行符。同时，它也支持基于 Character、Word、Line 等方式的对比。基于 Character 的对比方式会比较两个输入中每个字符的差异，适用于精细的差异检测。基于 Word 的对比方式用于比较两个输入中的每个单词，适合于检测文本内容的变化。基于 Line 的对比是按行比较数据，适用于处理结构化文本，如代码文件或日志。

当然 Diff 模块也能够对处理结果进行显示，它支持 Show added、Show removed、Show substraction、Ignore whitespace 共 4 种显示方式，例如，在 Input 面板中输入不同的数据，代码如下：

```
Hello World!
This is a test.


Hello World!
This is a different test.
```

如果在 Diff 模块中勾选 Show added 单选框并将 Diff by 参数设置为 Line，则会在 Output 面板中高亮显示 This is a different test 作为新增内容，如图 3-1 所示。

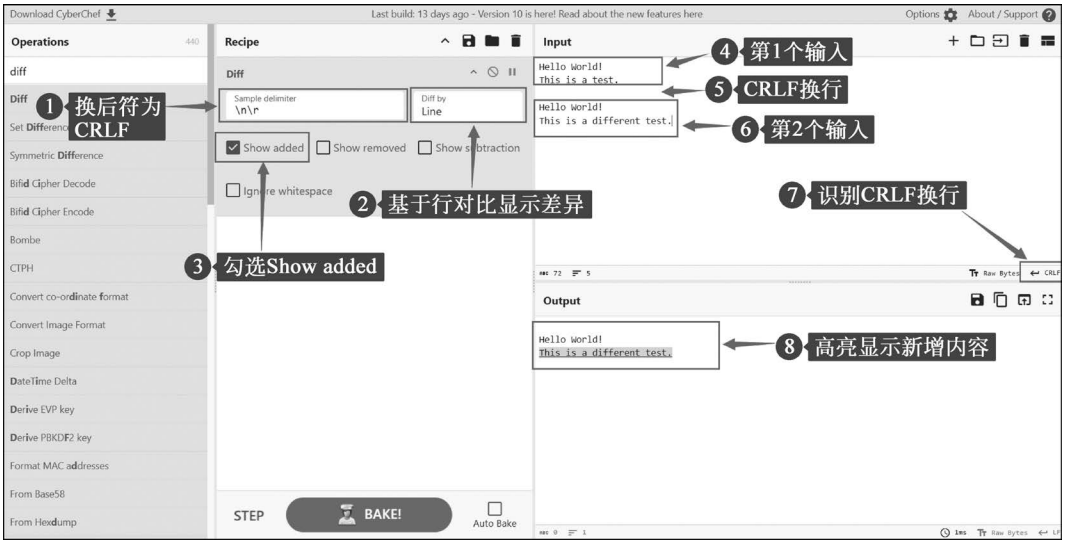


**图 3-1　使用 Diff 模块的 Show added 对比数据差异**

如果在 Diff 模块中勾选 Show removed 单选框并将 Diff by 参数设置为 Line，则会在 Output 面板中高亮显示 This is a test 作为减少的内容，如图 3-2 所示。

在 Diff 模块中，Show substraction 参数并不能单独使用。它需要结合 Show added 或 Show removed 参数共同使用，例如，如果同时勾选 Show added 和 Show substraction，则仅输出新增内容，如图 3-3 所示。

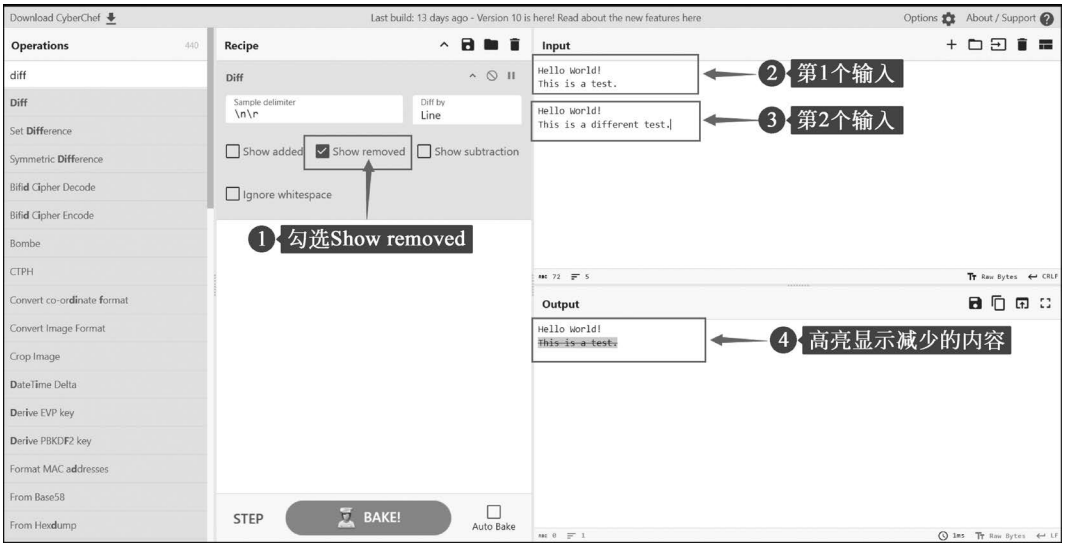


**图 3-2　使用 Diff 模块的 Show removed 对比数据差异**


**图 3-3　结合使用参数 Show added 与 Show substraction**

当然，如果用户同时勾选 Show removed 和 Show substraction，则只输出减少内容，如图 3-4 所示。

笔者认为所有的比较都是基于第 1 个输入内容，最终 CyberChef 会高亮显示它新增或减少的部分。

如果勾选 Ignore whitespace 单选框，则会在对比时忽略空格、制表符和换行符的差异。减少因为格式化、缩进或空格变化而导致的差异，使对比结果更关注实际内容的变化，而不是格式化上的差异。接下来，本书将以对比/etc/passwd 文件为例阐述关于 Diff 模块在实际环境中的使用方法。


**图 3-4　结合使用参数 Show removed 与 Show substraction**

Linux 操作系统中的/etc/passwd 是一个关键的系统文件，它用于存储用户账户信息。在/etc/passwd 文件中，每行代表一个用户账户，字段由冒号分隔。每个字段具有不同的含义，例如，在/etc/passwd 文件中，root 用户行对应的每个字段的含义如图 3-5 所示。

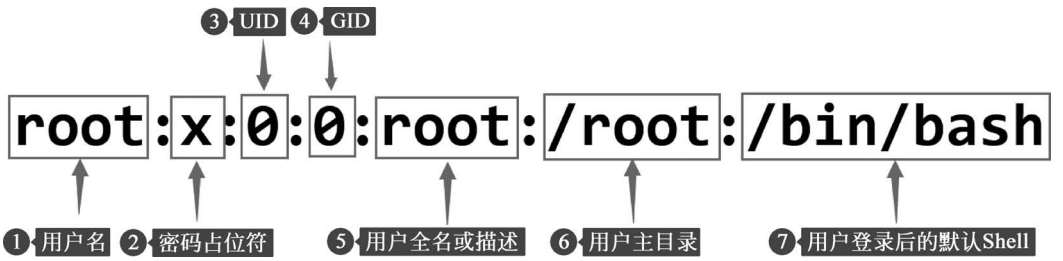


**图 3-5　/etc/passwd 内容行的字段含义**

在默认情况下，Linux 中的所有用户都可以读取/etc/passwd 文件内容，因此，该文件中采用密码占位符保存密码，而用户密码保存在/etc/shadow 文件中。如果黑客能够同时获取/etc/passwd 和/etc/shadow 文件，则可以利用这两个文件中的内容对 Linux 用户密码进行破解，因此，为了安全起见，在 Linux 系统中只有最高权限的 root 用户才可以读取/etc/shadow 文件内容。

Linux 操作系统将用户分为普通用户、系统用户、超级用户，其中，普通用户用于日常操作和应用程序的使用，每个普通用户都有一个唯一的 UID，通常 UID 是从 1000 开始的。这些用户不具有系统级别的权限。

系统用户用于系统服务和守护进程，UID 通常低于 1000。这些用户有时不需要登录系统，仅用于运行系统服务，例如，www-data、mysql 等用户。

超级用户通常是指 root 用户，它的 UID 为 0，具有系统的所有权限和控制权，可以执行任何操作。

当然，任何用户在/etc/passwd 文件中都具有对应的行，用于保存用户信息。

如果在 Linux 操作系统中新增用户，则会在/etc/passwd 文件中留有痕迹。那么，就可以使用 CyberChef 提供的 Diff 模块来对比新增用户前后的/etc/passwd 文件的内容差异，从而识别新增用户的信息，如图 3-6 所示。

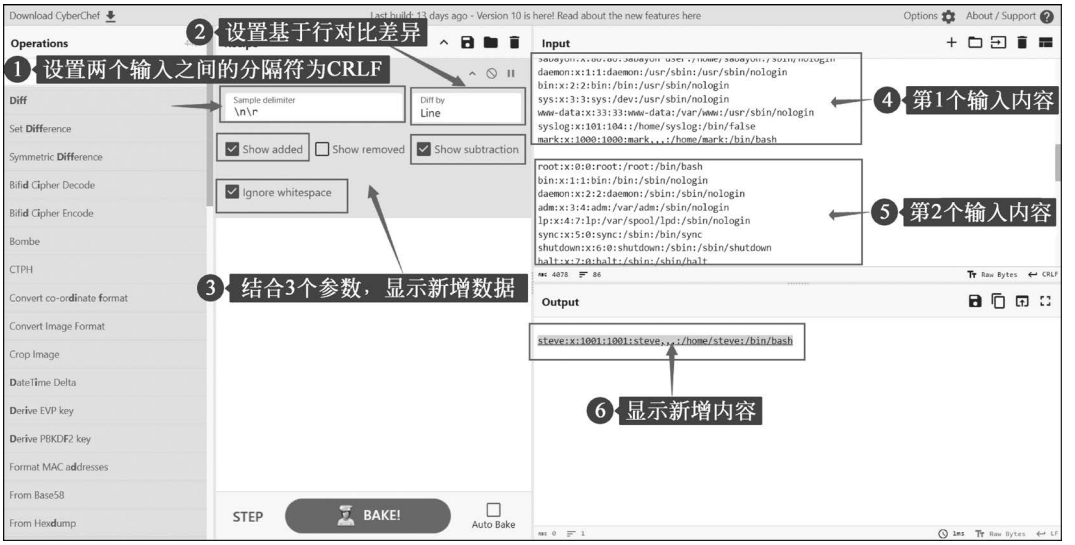


**图 3-6　对比两个输入的/etc/passwd 文件内容，显示新增行**

显然，使用 CyberChef 工具的 Diff 模块筛选出新增的用户名为 steve。当然，也可以尝试结合 Show removed 和 Show substration 参数来查看删除的用户信息。

## 3.1.2　提取数据

在数据处理过程中，提取数据是指从原始数据集中提取有价值的信息。它的目的通常是为了进一步分析、报告或决策。在 CyberChef 中的 Extractors 模块分组集成了多种提取数据的模块，用户使用这些模块能够快速地提取到相关的数据。

其中，Strings 模块用于提取输入数据的字符串信息，例如，在 Input 面板中加载可执行程序，并使用 Strings 模块来提取字符串，如图 3-7 所示。

Strings 模块提供了 Encoding 参数，此参数能够设定编码类型，Minimum length 参数用于设置字符串的最小长度，Match 参数可以指定匹配结果为 ASCII 或 Unicode 编码类型。

同样地，通过勾选 Display Total 单选框的方式来显示所有的字符串信息，勾选 Sort 单选框能够对结果进行排序，以及勾选 Unique 单选框实现对字符串进行去重复功能。

Extract IP addresses 模块能够提取输入数据中包含的 IP 地址信息，包括 IPv4 和 IPv6 地址。互联网协议地址(Internet Protocol Address，IP 地址)是用于标识网络中每台设备的唯一标识符。它就像是设备在互联网上的“地址”，使数据可以准确地发送到目标设备。IP 地址通常有两种格式，分别是 IPv4 和 IPv6 地址，其中，IPv4 地址是最常用的格式，由 4 组数字组成，每组数字的范围是 0 到 255，数字之间用点分隔，例如，192.168.0.1。


**图 3-7　使用 Strings 模块提取可执行文件的字符串信息**

由于全球互联网设备数量的增加，为了能够弥补地址数量不足的问题，IPv6 地址应运而生。IPv6 使用 128 位地址，通常以八组四个十六进制数字组成，并用冒号分隔，例如 2001:0db8:85a3:0000:0000:8a2e:0370:7334。

当然，IPv4 地址同样也进行了优化以解决地址不足的问题。这种方案是通过设定公有 IP 地址和私有 IP 地址来实现的。公有 IP 地址是由互联网服务提供商(Internet Service Provider，ISP)分配的，用于标识连接到互联网的设备。这种设备分配的 IP 地址可以被任意连接到互联网的计算机所访问，而私有 IP 地址用于局域网内部，不同局域网之间的私有 IP 地址可以重复。私有 IP 地址范围由 192.168.0.0 到 192.168.255.255、由 10.0.0.0 到 10.255.255.255、由 172.16.0.0 到 172.31.255.255。

Extract IP addresses 模块提供了 IPv4、IPv6、Remove local IPv4 addresses、Display total、Sort、Unique 单选框来满足提取 IP 地址的多种需求，例如，利用该模块来提取数据中包含的公有 IPv4 地址信息，如图 3-8 所示。

Extract email addresses 模块用于从文本中提取电子邮件地址。电子邮件地址是用于在互联网上发送和接收电子邮件的唯一标识符。它由两个主要部分组成，包括本地部分和域名部分，其中，本地部分表示收件人的用户名或标识符，而域名部分用于指定电子邮件服务器的地址，例如完整的电子邮件地址的格式为 username@domain.com，其中 username 为本地部分，domain.com 是域名部分。这个地址使邮件系统能够将发送的邮件准确地送到目标用户的邮箱。

Extract IP addresses 模块提供了 Display total、Sort、Unique 单选框，这些单选框能够满足提取电子邮件地址的各种需求。利用该模块来提取数据中包含的电子邮件地址信息，如图 3-9 所示。


图 3-8　使用 Extract IP addresses 模块提取 IPv4 公有 IP 地址


图 3-9　使用 Extract email addresses 模块提取电子邮件地址

当然，使用 Extract email addresses 模块提取邮件地址的过程中也可能会遭遇意外错误，例如，在数据中包含 www. baidu. com/img/flexible/logo/pc/result@2. png 等类似链接的情况，如图 3-10 所示。

这种错误的本质是由于该模块在提取电子邮件地址的过程中，只匹配电子邮件的格式，并未对其进行验证。那么只要在 URL 网址中添加@符号就会被匹配为电子邮件地址并对其进行输出。由于 Extract email addresses 模块是通过内置的正则表达式实现的，因此用户可以通过调用 Regular expression 模块来执行自定义正则表达式，从而正确地匹配电子


**图 3-10　Extract email addresses 模块的意外错误**

邮件地址。笔者常用正则表达式“\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}\b(?!\S)”来匹配电子邮件地址。关于正则表达式的相关内容，本书在 3.2.1 节中将会做详细说明。

Extract Urls 模块是一个简便且高效的工具，能够从网络流量、邮件、日志文件中快速地提取潜在的恶意网址，从而进一步地进行安全分析和应急响应。统一资源定位符(Uniform Resource Locator，URL)是一种用于在互联网上定位和访问资源的标准格式。URL 网址的使用带来了高效、标准化和用户友好的资源访问方式。它不仅简化了资源的定位和访问，还支持动态内容生成、数据传输、链接创建等功能。

URL 网址由 6 部分组成，包括协议、主机名、端口、路径、查询字符串、片段，如图 3-11 所示。

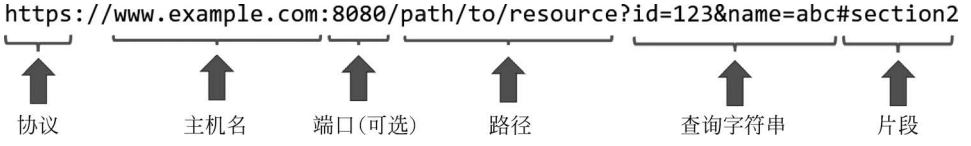


**图 3-11　URL 网址结构**

当然，组成 URL 网址结构的每部分都具有不同的功能，并且它们都可以设置不同的值，如表 3-1 所示。

**表 3-1　URL 网址结构的组成**

| URL 组成部分 | 功　能 |
| --- | --- |
| 协议 | 指示如何访问资源，支持 http://、https://、ftp://等 |
| 主机名 | 资源所在的服务器地址，例如，www.example.com |

续表

| URL 组成部分 | 功　　能 |
| --- | --- |
| 端口(可选) | 服务器上的具体端口号,HTTP 默认 80 端口,HTTPS 默认 443 端口 |
| 路径 | 资源在服务器上的具体位置 |
| 查询字符串(可选) | 传递参数的信息,通常以“?”开始,例如,id=123&name=abc |
| 片段(可选) | 指向文档中的特定部分,以“#”开始,例如,section2 |

Extract URLs 模块提供了 Display total、Sort、Unique 单选框,这些单选框能够满足提取 URL 网址的各种需求。利用该模块来提取数据中包含的 URL 网址信息,如图 3-12 所示。


**图 3-12　使用 Extract URLs 模块提取 URL 网址**

当然,读者也可以自行尝试使用 Extract Domain 模块来提取域名地址信息。在分析恶意程序样本时,提取的 URL 网址或域名极有可能包含恶意程序,因此,在完成提取后,笔者经常会调用 Defang URL 模块将潜在的恶意内容转换为安全的格式,以便进行安全审计和分析。

通过替换 URL 网址中的特殊字符,防止意外执行或触发,从而提高网络安全操作的安全性和有效性。Defang 支持 Escape dots、Escape http、Escap ://3 个参数,其中,参数 Escape dots 会将 URL 网址中的“.”号转换为[.]。参数 Escape http 可以把 http 转换为 hxxp。参数 Escape ://能够将://转换为[://],例如,调用 Defang URL 模块将 http://malware.com/test.exe 地址转换为安全格式,如图 3-13 所示。

在 Extractors 模块分组中,还提供了 Extract MAC addresses、Extract file paths、Extract dates 等模块。感兴趣的读者可以自行尝试使用这些模块提取数据。

### 3.1.3　格式化数据

数据格式化是将数据转换为一致且标准化的格式的过程。数据格式化不仅是数据处理


**图 3-13　使用 Defang URL 模块将 URL 网址转换为安全格式**

的基础步骤，也是确保数据质量和提高处理效率的重要环节。通过标准化数据格式，可以极大地简化后续的数据操作，提升数据分析的有效性，支持决策制定，并确保数据的可靠性和一致性。

CyberChef 提供了用于处理各种数据格式化任务的模块，通过这些模块，用户可以轻松地将数据转换为所需的格式，其中，CSV to JSON 模块可以将 CSV 格式数据转换为更灵活的 JSON 格式。

逗号分隔值(Comma-Separated Values，CSV)是一种简单的文件格式，用于存储表格数据。CSV 文件以纯文本形式存储数据，每行表示一条记录，每个字段之间使用逗号分隔。它被广泛地用于数据交换和存储，尤其是在数据导入和导出操作中。当然，CSV 格式因其简单、易用和广泛支持，成为数据处理和交换中的一种常见格式，例如，具有 Name、Age、City 这 3 个字段的 CSV 文件内容，代码如下：

```
Name,Age,City
John Doe,29,New York
Jane Smith,34,Los Angeles
Alice Johnson,27,Chicago
```

由于 CSV 只能表示平面的表格数据，无法直接处理复杂的数据结构，因此，在 Web 开发和 RESTful API 中常用 JSON 作为数据交换的格式，它与 JavaScript 的对象表示方式兼容，便于在前端应用程序中使用。同时，JSON 支持嵌套数据结构，能够更灵活地适应不同的需求和变化，表示复杂的层级关系和对象，例如，一个 JSON 对象可以包含嵌套的数组或其他 JSON 对象。

JSON 的灵活性和丰富的数据表示能力使其在与高级数据处理工具集成时非常有用。

如果使用 CSV to JSON 模块转换上述 CSV 格式的内容，则会获得对应的 JSON 数据，代码如下：

```
[
    {
        "Name": "John Doe",
        "Age": "29",
        "City": "New York"
    },
    {
        "Name": "Jane Smith",
        "Age": "34",
        "City": "Los Angeles"
    },
    {
        "Name": "Alice Johnson",
        "Age": "27",
        "City": "Chicago"
    }
]
```

当然，在 CSV to JSON 模块中提供了 Cell delimiters、Row delimiters、Format 这 3 个参数，其中，参数 Cell delimiters 用于设置 CSV 中字段的分隔符，默认字段分隔符为逗号。参数 Row delimiters 能够指定 CSV 中记录的分隔符，默认记录分隔符是\r\n，而参数 Format 是可以设定输出 JSON 数据的格式。默认格式为 Array of dictionaries。这种格式也是主流的 JSON 格式，如图 3-14 所示。

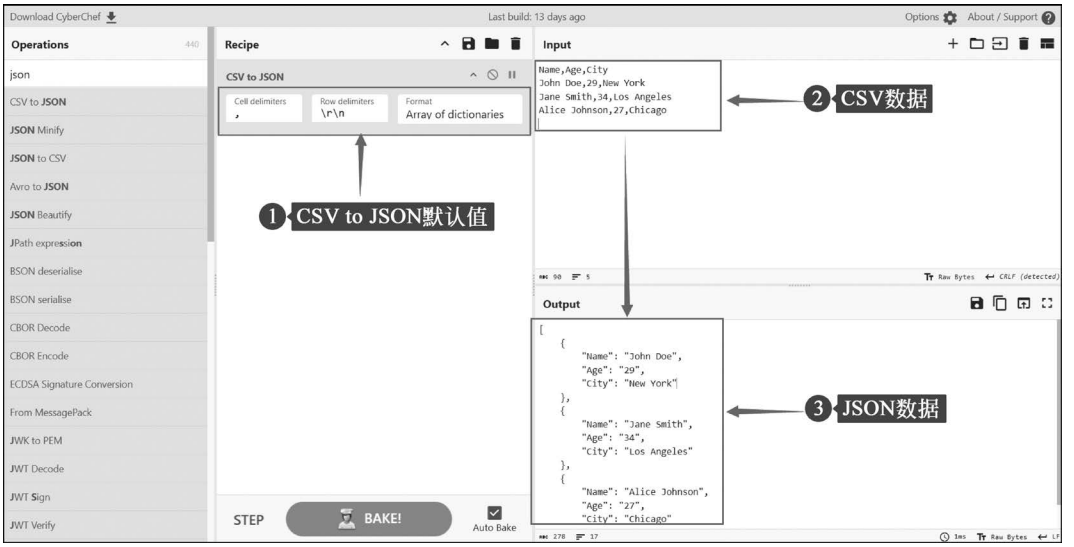


**图 3-14　CSV to JSON 模块转换格式**

格式化数据的本质是将同一数据在不同规定的格式中进行转换。CyberChef 中的 Change IP format 模块能够实现 IP 地址格式转换的功能。IP 地址可以用不同的进制表示，包括十进制、十六进制、八进制、二进制，例如，IP 地址 192.168.1.1 对应的十六进制、八进制、二进制格式的 IP 地址如表 3-2 所示。

**表 3-2　IP 地址 192.168.1.1 不同进制格式的 IP 地址**

| 进　　制 | IP 地址 |
|---|---|
| 十进制 | 192.168.1.1 |
| 十六进制 | C0.A8.01.01 |
| 八进制 | 300.250.1.1 |
| 二进制 | 11000000.10101000.00000001.00000001 |

在日常生活中，使用的 IPv4 地址经常是以点分十进制表示的。在这种格式中，IP 地址以 4 个十进制数表示，每个十进制数的范围为从 0 到 255，它们之间用点分隔。当然，将点分十进制 IP 地址转换中的每个十进制数转换为不同的数制就能够实现 IP 地址格式的变换，例如，可以将点分十进制 IP 地址 192.168.1.1 转换为点分十六进制 IP 地址 C0.A8.01.01。

不同数制格式的 IP 地址，本质上都是从二进制格式的 IP 地址转换而来的。Change IP format 模块提供了 Input format 和 Output format 两个参数。参数 Input format 用于指定输入数据的 IP 地址格式，而参数 Output format 能够设定输出数据的 IP 地址格式，例如，使用该模块将十进制格式的 IP 地址 192.168.1.1 转换为十六进制格式的 IP 地址，如图 3-15 所示。

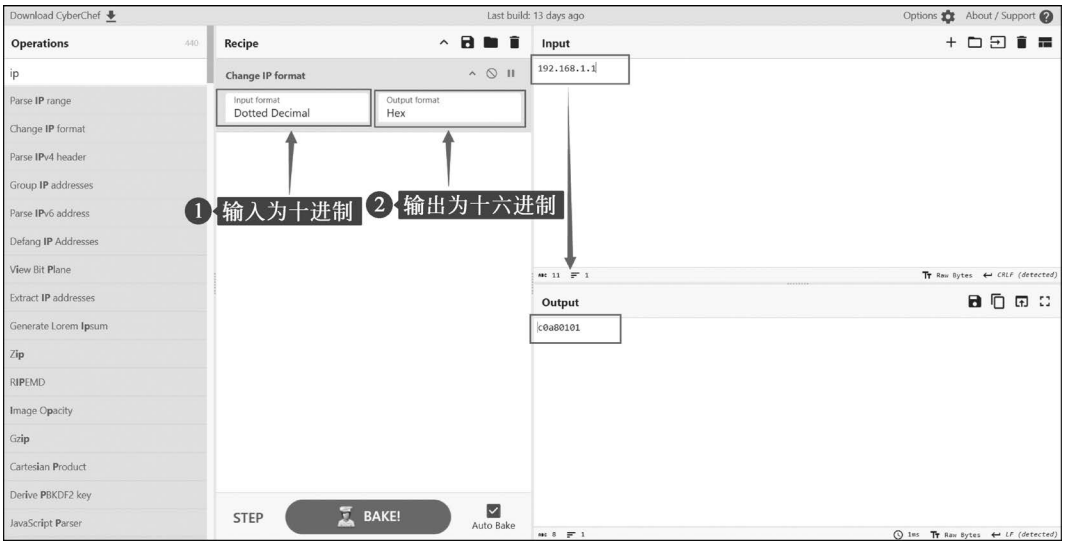


**图 3-15　转换 IP 地址进制格式**

细心的读者会发现使用 Change IP format 模块转换的 IP 地址格式不存在点号。在 IP 地址中的点号是人为设定的，因此去除点号并不会影响正常使用 IP 地址。当然，格式化数

据不仅为了转换格式，同样也需要美化数据格式以便于查看，例如，Code tidy 模块分组中提供了许多用于美化数据的模块，其中，Syntax highlight 模块能够高亮显示输入的数据，如图 3-16 所示。


**图 3-16　高亮显示输入的数据**

Syntax highlighter 模块提供的 Language 参数用于设置需要识别的输入数据的语言类型，默认的 auto detect 能够自动识别对应的语言类型。虽然该模块能够根据编程语言的类型高亮显示语法结构，但是它并不能对压缩过的程序代码进行美化处理，例如，将经过压缩处理的 JavaScript 代码输入 CyberChef 的 Input 面板，并调用 Syntax highlighter 模块进行处理，如图 3-17 所示。

压缩 JavaScript 代码是前端开发中的一个重要步骤，其主要目的是优化网页性能和提升用户体验。使用压缩后的 JavaScript 代码能够减小加载时间、节省带宽，从而优化用户体验。当然，使用压缩代码也能够提高代码安全性。压缩过程通常包括代码混淆，即将代码中的变量名、函数名等替换为不易理解的短名称，从而增加逆向工程的难度。虽然这不是一种完全的安全措施，但可以增加攻击者的分析难度。

为了能够更清晰地查看 JavaScript 代码，用户可以调用 JavaScript Beautify 模块来美化压缩代码，如图 3-18 所示。

当然，笔者会结合 JavaScript Beautify 和 Syntax highlighter 两个模块来美化 JavaScript 的压缩代码。

虽然，CyberChef 中提供了许多用于格式化的模块，但是本书仅设置了部分常用的模块。感兴趣的读者可以根据本书介绍的模块使用方法来尝试使用其他格式化相关模块。

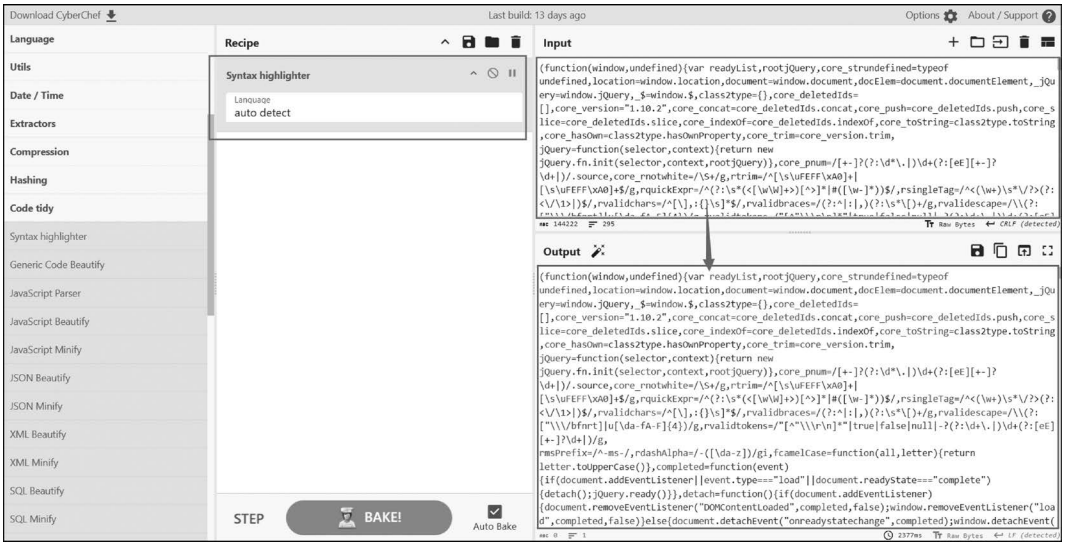


图 3-17 Syntax highlighter 模块不进行代码美化处理


图 3-18 美化 JavaScript 压缩代码

## 3.2 数据的匹配操作

数据匹配操作的本质是通过正则表达式等模式匹配技术对输入数据进行比对，以提取符合特定模式或结构的信息，从而实现数据的筛选、整理和整合。通过有效的数据匹配，用户可以清除重复记录、整合来自不同来源的信息、验证数据质量，并提高分析结果的可靠性，

从而支持更精准的决策。在恶意程序样本分析领域中，笔者经常使用数据匹配操作来提取相关数据。

### 3.2.1　介绍正则表达式

正则表达式是一种用于匹配、搜索和操作字符串的工具，它使用特定的语法规则定义模式。正则表达式的模式也常直接被称为正则表达式。通过正则表达式定义的模式，能够对数据进行匹配，并得到匹配结果，如图 3-19 所示。

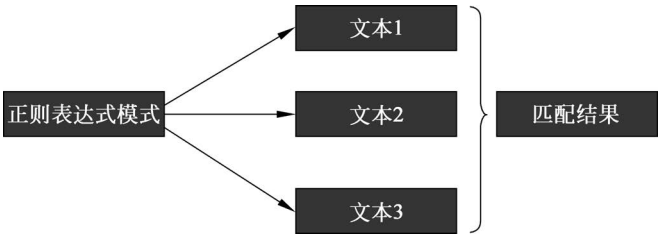


**图 3-19　正则表达式匹配数据的基本原理**

正则表达式并非特定编程语言的专有工具。它是一种通用的文本处理技术，被广泛地应用于多种编程语言和工具中，例如 Python、JavaScript、Java、Perl、grep 等。不同语言可能有其特定的正则表达式和语法细节，但基本概念和功能是一致的。CyberChef 工具提供的 Regular expression 模块能够将自定义的正则表达式模式应用到 Input 面板中输入数据，并在 Output 面板中输出对应的匹配结果，因此，笔者经常使用该模块来匹配数据。

为了能够充分地使用 Regular expression 模块的功能，需要用户具有编写正则表达式的能力。正则表达式本质上是能够匹配数据的字符串，它是由特定的字符组成的。这些字符包括元字符、预定义字符、量词、分组字符、选择字符、转义字符等，如表 3-3 所示。

**表 3-3　正则表达式的组成字符**

| 字　　符 | 功　　能 |
|---|---|
| . | 匹配任意单个字符，除换行符外 |
| ^ | 匹配字符串的开头 |
| $ | 匹配字符串的结尾 |
| * | 匹配前面的字符零次或多次 |
| + | 匹配前面的字符一次或多次 |
| ? | 匹配前面的字符零次或一次 |
| [abc] | 匹配字符 a、b 或 c |
| [^abc] | 匹配除 a、b 和 c 之外的任意字符 |
| [a-z] | 匹配任意小写字母 |
| [A-Z] | 匹配任意大写字母 |
| [0-9] | 匹配任意数字 |
| \d | 匹配任意数字，等同于[0-9] |

续表

| 字　符 | 功　能 |
| --- | --- |
| \D | 匹配任意非数字字符 |
| \w | 匹配任意字母、数字字符和下画线，等同于[a-zA-Z0-9_] |
| \W | 匹配任意非字母、数字字符和下画线 |
| \s | 匹配任何空白字符，例如，空格、制表符、换行符 |
| \S | 匹配任何非空白字符 |
| {n} | 匹配前面的字符恰好 *n* 次 |
| {n,} | 匹配前面的字符至少 *n* 次 |
| {n,m} | 匹配前面的字符至少 *n* 次，但不超过 *m* 次 |
| () | 用于分组，并捕获匹配的子字符串，例如，(abc)匹配 abc 并将其捕获 |
| (?:) | 用于分组，但不捕获匹配的子字符串。用于更复杂的模式匹配而不保存结果 |
| \| | 表示“或”操作，匹配符号前后的任意模式，例如，a\|b 匹配 a 或 b |
| \ | 转义特殊字符，使其作为普通字符进行匹配，例如，\. 用于匹配一个实际的点(.)，而不是任意字符 |

编写正则表达式需要对其语法和特性有充分的理解，以便构建精确的匹配模式，例如，编写匹配电子邮件的正则表达式，代码如下：

```
^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$
```

其中，符号^和$分别表示正则表达式的开头和结尾部分。“^[a-zA-Z0-9._%+-]+”用于匹配用户名部分，“@[a-zA-Z0-9.-]+”用于匹配域名部分，“\.[a-zA-Z]{2,}$”用于匹配域后缀部分。

用户可以将 Regular expression 模块的 Regex 参数设置为上述正则表达式，用来匹配输入数据中的电子邮件地址。当然，在 Regular expression 模块的 Built in regexes 参数中默认内置了许多正则表达式，包括 IPv4 address、URL、Domain、Email address 等。用户可以根据需求选择相应的正则表达式。当然，内置的正则表达式并不能满足所有匹配需求，因此，掌握编写正则表达式的能力是势在必行的，例如，在恶意样本文件中，具有许多 Base64 编码的字符串。如果不使用正则表达式进行匹配，则直接通过手工解码的方式将会是一项非常具有挑战性的任务。

Regular expression 模块具有 Built in regexes、Regex、Output format 等参数，其中，参数 Built in regexes 用于设置正则表达式的类型。如果用户需要使用自定义正则表达式，则必须将该参数设置为默认的 User defined。参数 Regex 是填写正则表达式的文本框。如果将 Built in regexes 参数设置为 User defined，则 Regex 的值需要用户填写自定义的正则表达式，否则该模块会自动填充内置的正则表达式。参数 Output format 用于指定输出结果的格式，它的值包括 Highlight matches、List matches 等。Highlight matched 能够在 Output 面板中高亮显示匹配结果，List matched 则可以列出所有匹配的结果。接下来，本书将介绍使用用户自定义正则表达式匹配样本文件中的 Base64 编码，并对其进行解码

输出。

首先，构建匹配 Base64 编码的正则表达式，代码如下：

```
[0-9a-zA-Z+/=]{50,}
```

在上述代码中，[0-9a-zA-Z+/=]用于限定 Base64 编码的字符范围，{50,}用于设定至少匹配 50 个字符才能确定为 Base64 编码。当然，由于实际的样本文件中极有可能存在混淆字符串，因此可以通过放大或缩小匹配字符的个数来绕过混淆字符。

接下来，在 CyberChef 中调用 Regular expression 模块，将参数 Built in regexes 的值设置为[0-9a-zA-Z+/=]{50,}，将 Output format 的值设置为 List matches。同时，将样本输入填写到 Input 面板中。如果样本文件中存在匹配结果，则会在 Output 面板中输出结果，如图 3-20 所示。

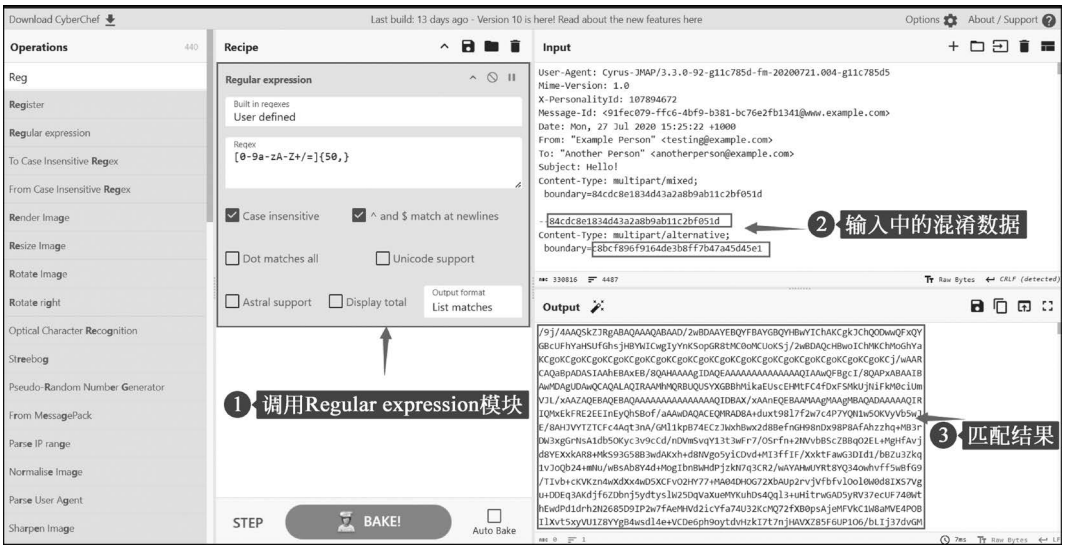


**图 3-20　匹配样本中的 Base64 编码字符串**

最后，通过调用 From Base64 模块实现对 Base64 编码字符串的解码，如图 3-21 所示。

细心的读者会发现在 Output 面板的输出结果中，具有 JFIF 字符串，它表明这是一个图片文件。

JFIF 是一种用于存储和交换 JPEG 图像的文件格式。它是一种标准的图像文件格式，用于确保不同设备和软件能够正确地解码和显示 JPEG 图像。JFIF 文件通常以.jpg 或.jpeg 扩展名保存，因此，用户可以使用 CyberChef 提供的 Detect File Type 模块来识别结果数据对应的文件类型。Detect File Type 模块是根据文件的内容而不是文件扩展名来确定文件格式的，这样便可确保文件类型的准确识别，有助于进一步地进行分析或处理。

如果使用 Detect File Type 模块成功地识别了文件类型，则会在 Output 面板中输出文件信息，如图 3-22 所示。

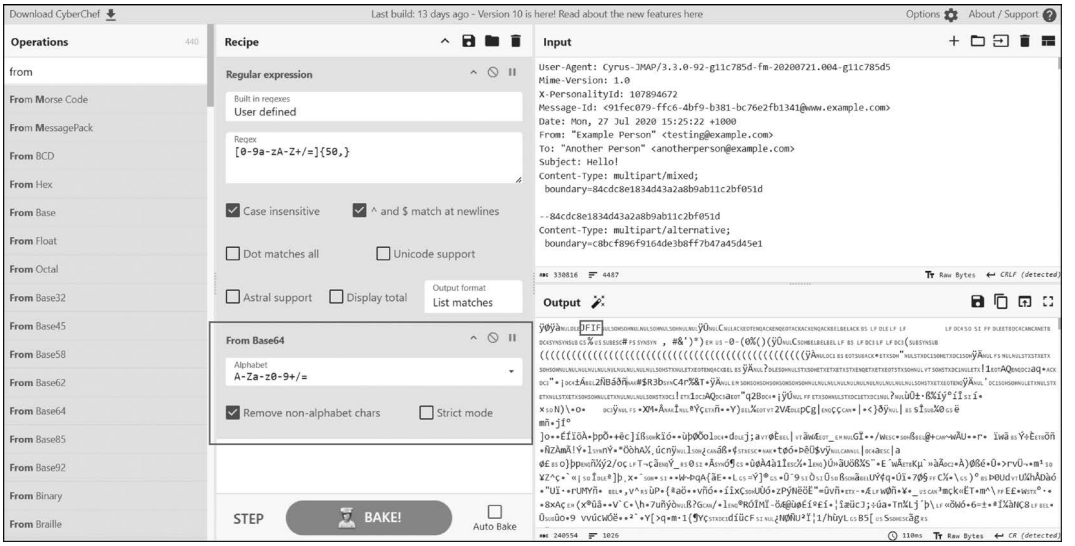


**图 3-21　使用 From Base64 模块进行解码**

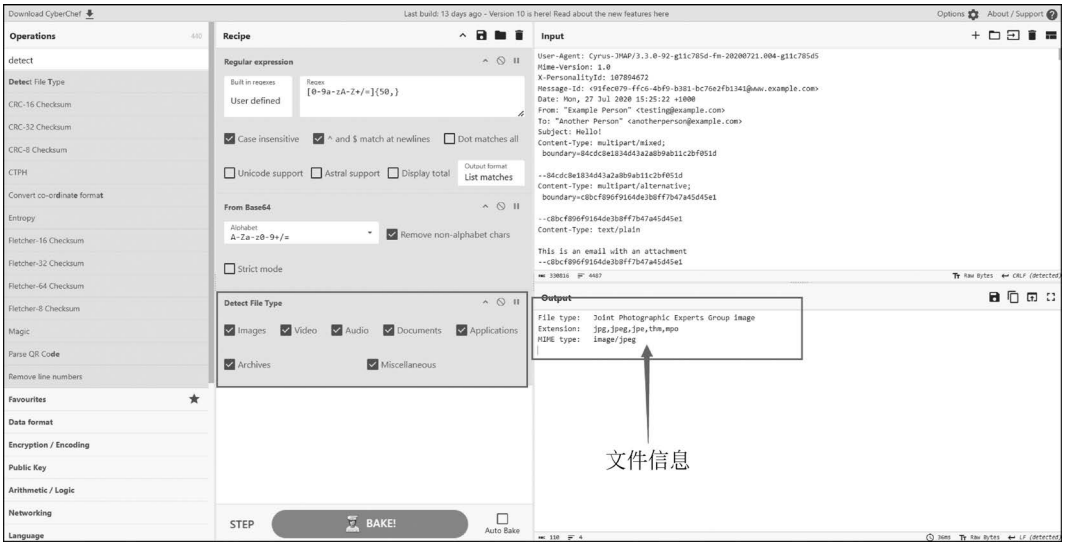


**图 3-22　Detect File Type 模块输出文件信息**

根据 Detect File Type 模块的检测结果表明该文件为图片文件，因此，用户可以通过调用 CyberChef 的 Render Image 模块来渲染并输出图片文件，如图 3-23 所示。

由于本书并非关于正则表达式的书籍，因此本章节仅介绍部分关于正则表达式的知识，因此，读者可以自行查阅资料学习更多关于正则表达式的内容。

## 3.2.2　分析日志文件

日志文件是记录系统、应用程序或服务活动的文本文件。通过分析日志文件能够迅速

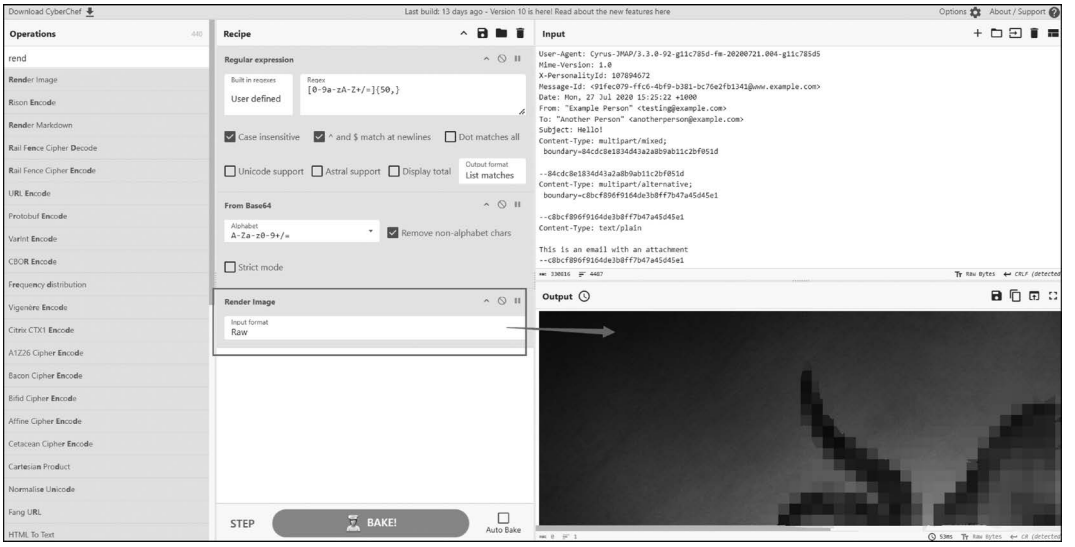


**图 3-23　Render Image 模块渲染图片文件**

地识别系统故障或应用程序错误的原因，因此，分析日志是确保系统稳定性、安全性和性能的关键步骤，是现代运维和数据分析的重要组成部分。接下来，本书将以分析 Web 日志文件为例来阐述关于分析日志的方法。

Web 日志文件是记录 Web 服务器活动的日志文件。它们通常用于跟踪和分析 Web 服务器的请求和响应情况，帮助管理员和开发人员进行故障排查、性能优化和安全分析。

常用的 Web 服务器软件包括 Nginx、Apache 等。接下来，本书将以 Nginx 的访问日志文件为例说明使用 CyberChef 工具分析日志文件内容的方法。

Nginx 是一个功能强大且灵活的 Web 服务器和代理服务器，被广泛地应用于各种规模的企业和网站。它的高性能、低资源消耗和强大的配置能力使其成为现代 Web 基础设施的核心组件。

Nginx 日志文件通常可分为访问日志、错误日志，其中，访问日志的默认保存路径为 /var/log/nginx/access.log，这个日志文件记录了每个客户端请求的详细信息。错误日志的默认路径是/var/log/nginx/error.log，它记录了服务器在处理请求过程中发生的错误，包括配置错误、运行时错误等。

用户可以通过分析 Nginx 访问日志文件中记录的内容来分析网站访问情况、排查问题和提升性能。访问日志文件记录的字段及其含义如表 3-4 所示。

**表 3-4　访问日志文件记录的字段及其含义**

| 字　　段 | 含　　义 |
| --- | --- |
| 时间戳 | 请求发生的日期和时间 |
| IP 地址 | 发起请求的客户端地址 |
| 请求方法 | HTTP 请求方法，例如，GET、POST 等 |

续表

| 字　　段 | 含　　义 |
| --- | --- |
| URL | 请求的资源路径 |
| 状态码 | 服务器响应的状态码 |
| 响应大小 | 返回的数据字节数 |
| 用户代理 | 客户端的浏览器和操作系统信息 |
| 来源页面 | 用户是从哪个页面跳转过来的 |

日志文件本质上是文本文件，使用任意文本编辑器都可以查看日志文件。在 Nginx 访问日志文件中，记录的每行都表示一个客户端请求，如图 3-24 所示。


**图 3-24　Nginx 访问日志文件的内容**

其中，192.168.1.100 为发起请求的客户端的 IP 地址，[28/Sep/2020:10:38:52 +0000]是客户端发送请求发生的日期和时间，GET 为客户端的请求方法，/login.php 是客户端的请求页面，HTTP/1.1 为客户端请求的 HTTP 协议版本，200 为服务器端响应的状态码，1191 是服务器端响应的字节长度，"-"表示不存在来源页面，客户端直接访问 login.php 页面，Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0 表明发送请求客户端使用的浏览器和操作系统的相关版本信息。

由于 Nginx 访问日志文件中保存着所有客户端访问 Nginx 服务器的访问记录，因此这个文件可能会变得特别大。如果用户直接使用文本编辑器通过手工的方式来分析访问日志文件，则会使分析任务变得非常困难，因此，借助 CyberChef 工具分析访问日志文件能够帮助分析人员提升效率。

首先，将访问日志文件加载到 CyberChef 工具的 Input 面板中，如图 3-25 所示。

如果 CyberChef 成功地加载了访问日志文件，则会在 Output 面板同样输出日志文件的内容。在分析访问日志文件时，分析人员需要特别关注的字段是发送访问请求的客户端 IP 地址、客户端的请求页面及客户端所使用的浏览器和操作系统相关的版本信息。

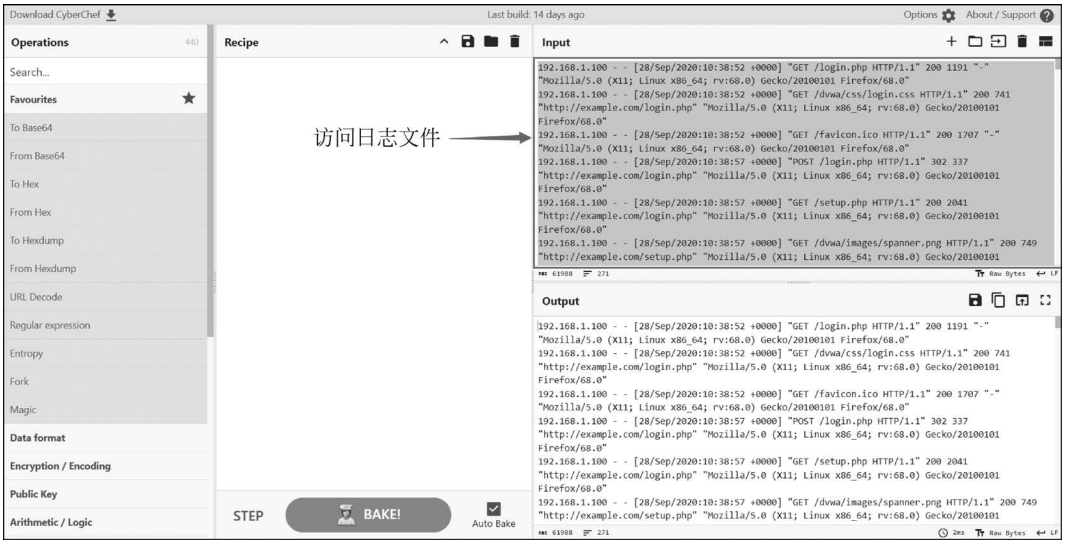


**图 3-25　加载访问日志文件**

接下来，通过调用 CyberChef 工具的 Extract IP addresses 模块来提取访问日志文件中的 IP 地址信息，如图 3-26 所示。

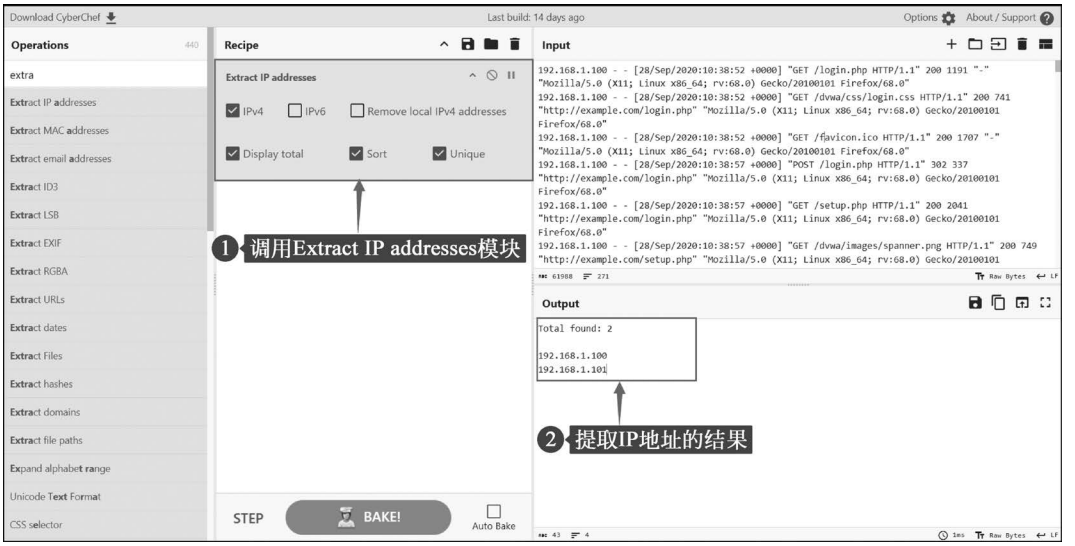


**图 3-26　执行 Extract IP addresses 模块**

在执行 Extract IP addresses 模块后，在 Output 面板中会输出访问日志文件中所具有的 IP 地址信息，包括 192.168.1.100 和 192.168.1.101 共两个 IP 地址。根据提取的 IP 地址，分析人员可以调用 Filter 模块并使用自定义正则表达式来过滤出包含 IP 地址的记录行信息。Filter 模块提供了 Delimiter、Regex、Invert condition 共 3 个参数，其中，Delimiter 参数用于设置分隔符，它可以设定 Input 面板中输入的分隔，例如，line feed 能够以行为单位

分隔输入数据。Regex 参数能够设置过滤输入数据的正则表达式。如果输入的数据匹配正则表达式，则会将对应记录输出到 Output 面板中。与此相反，参数 Invert condition 能反转 Regex 参数指定的条件。如果勾选了这个参数，则会在 Output 面板中输出不匹配的记录，例如，使用 Filter 模块过滤出输入数据中包含 192.168.1.100 的记录行，如图 3-27 所示。

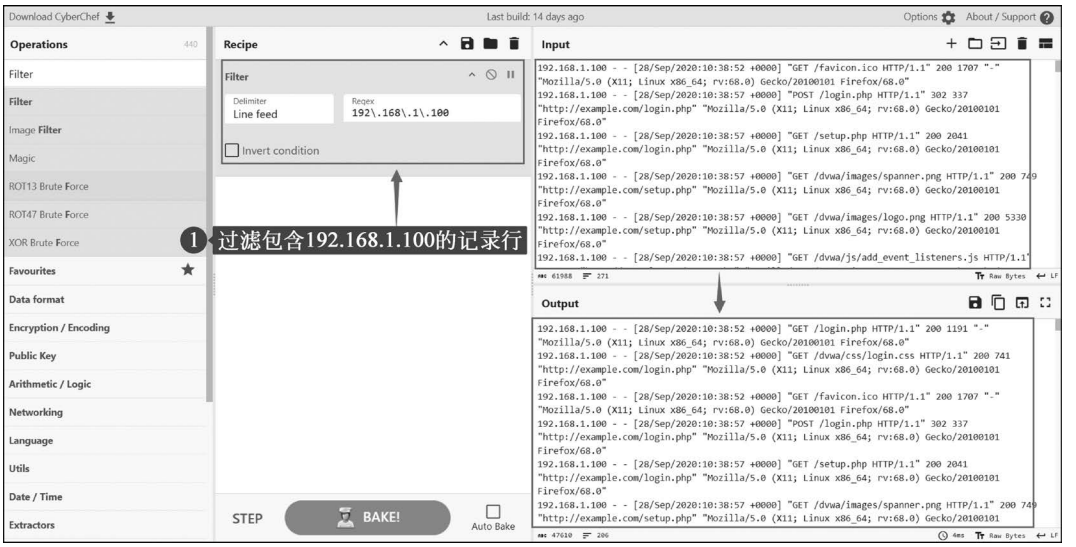


**图 3-27　执行 Filter 模块**

接下来，查看 Output 面板的内容，在记录行中会显示客户端 sqlmap，如图 3-28 所示。

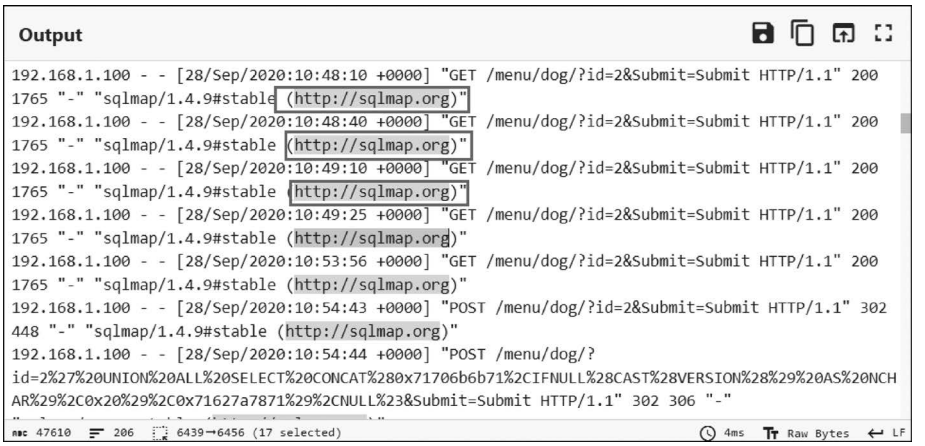


**图 3-28　分析 Output 面板内容**

为了能够查看包含 sqlmap 字符串的请求数量，可以通过调用 Count occurrences 模块来计算它的数量，如图 3-29 所示。

在 CyberChef 工具调用 Count occurrences 模块后，获知在访问日志文件中共 350 个请求具有 sqlmap。sqlmap 是一个开源的自动化 SQL 注入工具，旨在帮助安全专家和渗透测

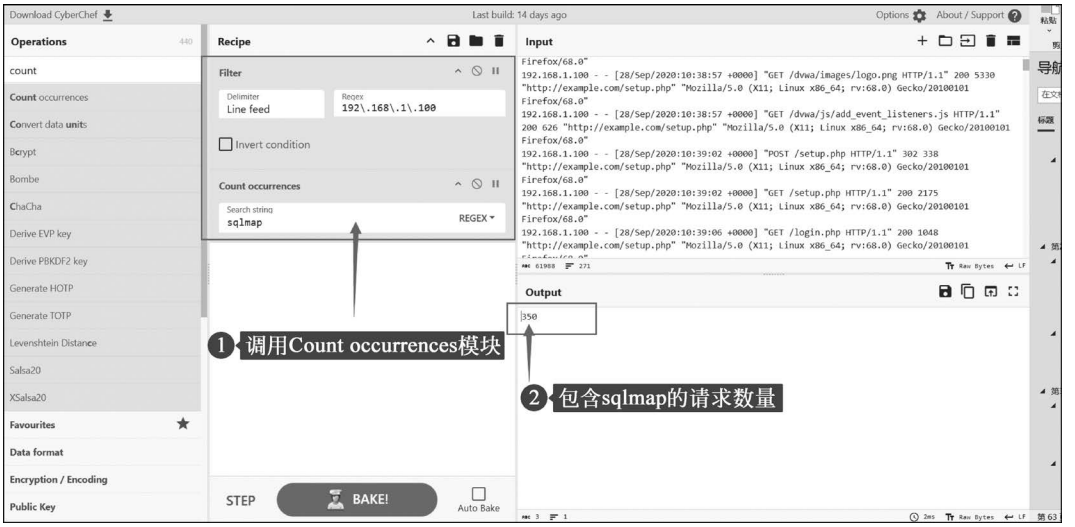


**图 3-29　执行 Count occurrences 模块**

试人员发现和利用 SQL 注入漏洞。它支持多种数据库管理系统，并提供了全面的功能，如自动化注入、数据提取和数据库管理。sqlmap 默认使用包含 sqlmap 字符串的客户端来访问 Web 应用程序。由此可见，攻击者可能会使用 sqlmap 工具对 Web 应用程序发起 SQL 攻击。

接下来，将 Filter 模块的 Regex 参数优化为 192\.168\.1\.100.＋sqlmap.*，在 Output 面板中输出包含 sqlmap 工具的记录行信息，如图 3-30 所示。

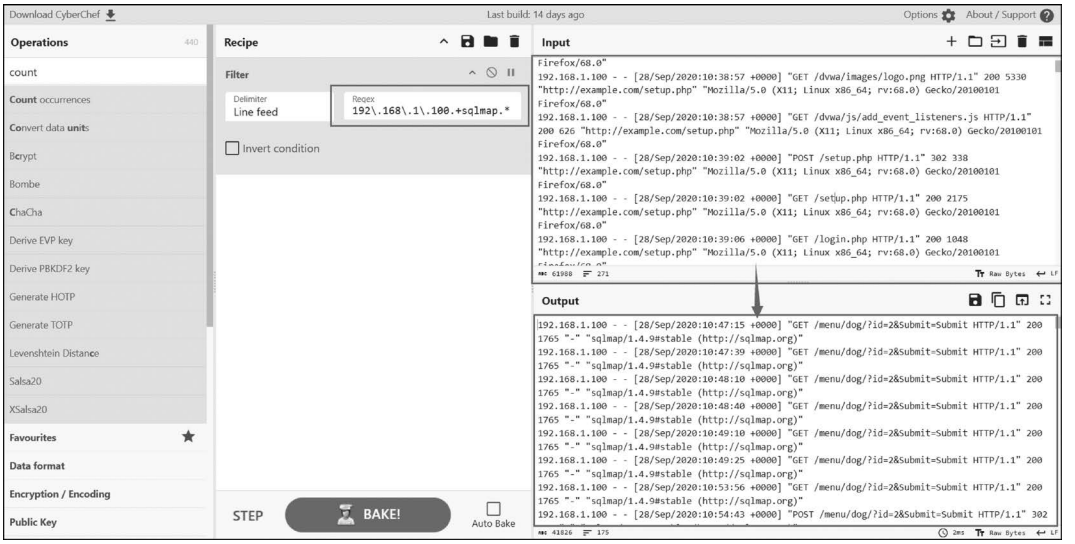


**图 3-30　优化 Filter 模块 Regex 参数**

为了更加直观地查看 Output 面板的信息，可以单击全屏显示按钮来显示输出数据，如图 3-31 所示。

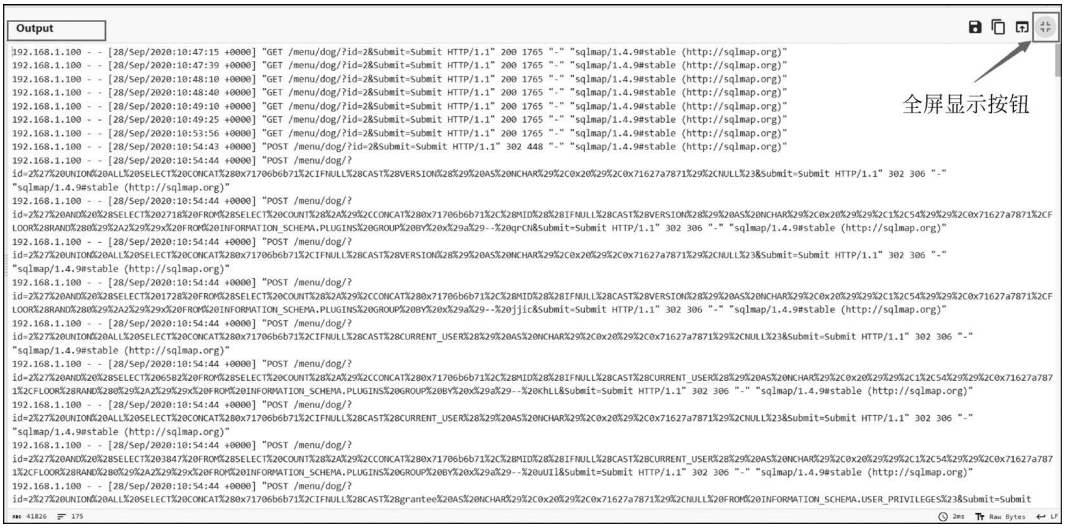


图 3-31　全屏显示输出数据

细心的读者会发现在 Output 面板的输出数据中，包含具有 URL 编码的数据。接下来，通过调用 URL Decode 模块来解码输出数据中的 URL 编码，如图 3-32 所示。

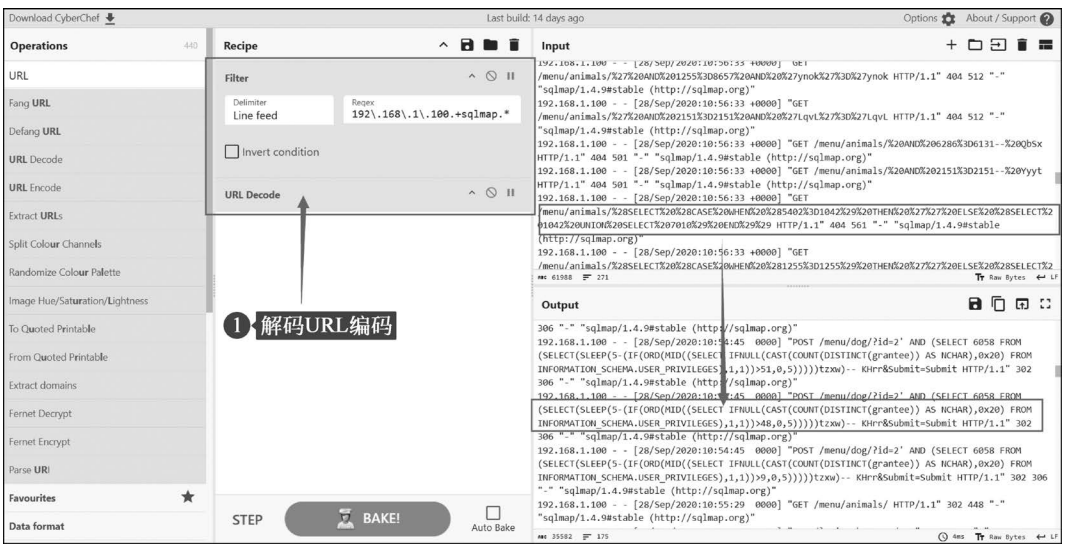


图 3-32　执行 URL Decode 模块

最后，通过分析请求记录中是否包含状态码 200 来确定攻击者是否使用 sqlmap 工具成功地执行了 SQL 注入攻击。笔者经常会再次调用 Filter 模块过滤出具有状态码 200 的数据，如图 3-33 所示。

在过滤的结果中，仅包含用于测试请求页面是否能够正常连接的记录，代码如下：

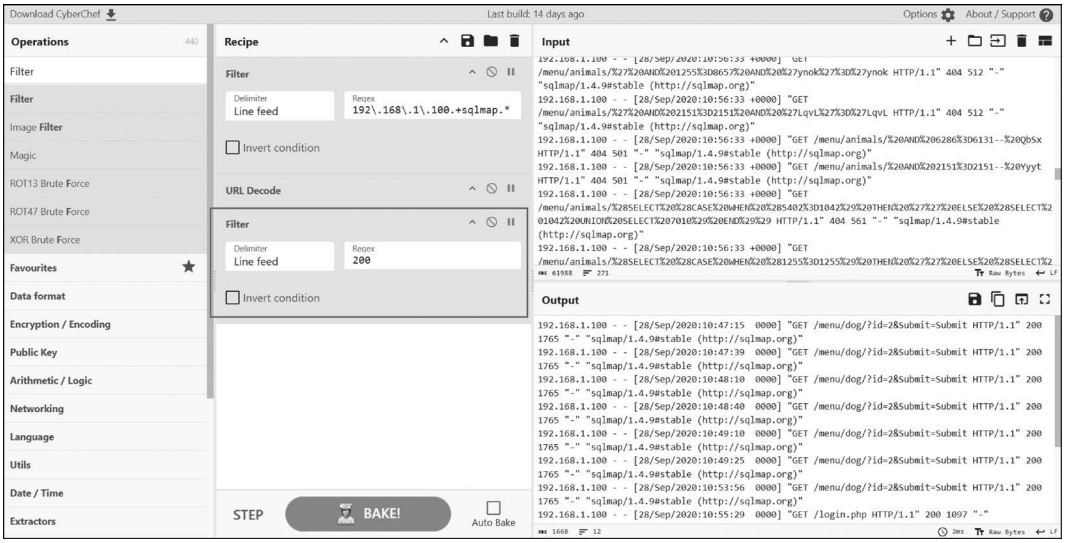


图 3-33　执行 Filter 模块过滤包含状态码 200 的记录

```
192.168.1.100 - - [28/Sep/2020:10:47:15 0000] "GET /menu/dog/?id=2&Submit=Submit HTTP/
1.1" 200 1765 "-" "sqlmap/1.4.9#stable (http://sqlmap.org)"
192.168.1.100 - - [28/Sep/2020:10:47:39 0000] "GET /menu/dog/?id=2&Submit=Submit HTTP/
1.1" 200 1765 "-" "sqlmap/1.4.9#stable (http://sqlmap.org)"
192.168.1.100 - - [28/Sep/2020:10:48:10 0000] "GET /menu/dog/?id=2&Submit=Submit HTTP/
1.1" 200 1765 "-" "sqlmap/1.4.9#stable (http://sqlmap.org)"
192.168.1.100 - - [28/Sep/2020:10:48:40 0000] "GET /menu/dog/?id=2&Submit=Submit HTTP/
1.1" 200 1765 "-" "sqlmap/1.4.9#stable (http://sqlmap.org)"
192.168.1.100 - - [28/Sep/2020:10:49:10 0000] "GET /menu/dog/?id=2&Submit=Submit HTTP/
1.1" 200 1765 "-" "sqlmap/1.4.9#stable (http://sqlmap.org)"
192.168.1.100 - - [28/Sep/2020:10:49:25 0000] "GET /menu/dog/?id=2&Submit=Submit HTTP/
1.1" 200 1765 "-" "sqlmap/1.4.9#stable (http://sqlmap.org)"
192.168.1.100 - - [28/Sep/2020:10:53:56 0000] "GET /menu/dog/?id=2&Submit=Submit HTTP/
1.1" 200 1765 "-" "sqlmap/1.4.9#stable (http://sqlmap.org)"
192.168.1.100 - - [28/Sep/2020:10:55:29 0000] "GET /login.php HTTP/1.1" 200 1097 "-"
"sqlmap/1.4.9#stable (http://sqlmap.org)"
192.168.1.100 - - [28/Sep/2020:10:55:29 0000] "GET /login.php HTTP/1.1" 200 1014 "-"
"sqlmap/1.4.9#stable (http://sqlmap.org)"
192.168.1.100 - - [28/Sep/2020:10:56:30 0000] "GET /login.php HTTP/1.1" 200 1096 "-"
"sqlmap/1.4.9#stable (http://sqlmap.org)"
192.168.1.100 - - [28/Sep/2020:10:56:33 0000] "GET /login.php HTTP/1.1" 200 1013 "-"
"sqlmap/1.4.9#stable (http://sqlmap.org)"
192.168.1.100 - - [28/Sep/2020:10:56:33 0000] "GET /login.php HTTP/1.1" 200 1015 "-"
"sqlmap/1.4.9#stable (http://sqlmap.org)"
```

根据上述请求记录，攻击者并未成功执行 SQL 注入攻击。虽然使用 CyberChef 能够快速地分析日志文件，但是它仅适用于文件大小适中的分析任务。对于极大的日志文件，笔者建议使用其他工具来对日志文件进行分析。