ManageEngine
DataSecurity Plus

How to **write Regex to find sensitive data**

www.datasecurityplus.com

Find sensitive personal data stored in your file servers using regular expressions or by matching keywords. Read on to learn what a regex is and how it can be used to find sensitive data.

What is a regex?

Regex is a string of alphanumeric and special characters used to search for and extract similar text patterns from log files, code, and other documents. In DataSecurity Plus, we use regex to find sensitive personal data such as PCI, PII, or ePHI.

Example: lisa@test.com (Email address) ^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$

How to write a regex?

Any single literal character is a regex by itself.

Note: For easier understanding, kindly note that the regex are highlighted in green and the phrase for which regex is written will be highlighted in blue.

Example: The regex D and the regex sec matches the phrase Data security.

Whereas regex S does not match the above phrase since regex is case-sensitive.

There are 12 characters with special meaning in regular expressions. These special characters when used alone are construed as errors. When you need to use a special character as a part of your regex, it needs to be preceded with a '\'.

Example: Regex for Data.security is Data\.security

Special character	Meta characters	Intended meaning	Examples
Caret	^	It is the anchor for the character at the start of the string.	The regex ^d will match any string that starts with 'd' such as data or deer
Dollar sign	$	It is the anchor for the character at the end of the string.	The regex a$ will match any string that ends with 'a' such as data or alpha
Vertical bar	|	It is used to separate a set of alternatives.	The regex dat(a|e|o) will match data, date, and dato.

Period	.	It represents any single character.	The regex Dat. will match both Date and Data
Question mark	?	It represents the optional part of the string.	The regex Jan(uary)? will match both Jan and January
Asterisk	*	It matches zero or more of the character preceding it.	The regex Dat*a will match both Datta and Daa
Plus	+	It matches one or more of the character preceding it.	The regex Dat+a will match both Datta and Dattta but not Daa
Parenthesis	()	It groups characters in a regex. Its use can also be recalled later using shorthand i.e., (\1).	The regex Dat(a|e) will match both Date and Data. The regex Jan(uary)[-]Febr(\1) will match January-February
Square brackets	[]	It represents any character inside. If ^ is used along with square brackets represents any character but the one mentioned.	The regex Dat[a-f] will match both Data and Date but not Dato. The regex Dat[^e] will match both Data and Dato but not Date.
Curly braces	{}	It is used to quantify the range.	The regex Data{2,3} will match Dataa and Dataaa
Minus sign	-	It is used to indicate the range of characters allowed.	The reg Dat[a-z] will match Data and Date i.e., Dat followed by any lowercase characters from a to z.
Backslash	\	It gives a special meaning to the character following it.	\n stands for new line; \w depicts a character. The regex \w will match d, g, p, and more.

Example: Consider the regex Info(rmation security?|(\ssec|sec)(urity|urities)?). It will find all variations of the expression 'Information security' from any document.
It will find the 4 different iteration's, i.e., Information security, Infosec, Info security, and Info securities.

Since special meta characters make the process of constructing regular expression easier and faster, it's imperative to understand their usage. Given below is the list of meta characters commonly used:

Special meta-character	Intended meaning	Examples
\d	It denotes any whole number (0-9).	The regex \d\d will match both 79 and 30 but not 5.
\w	It denotes any alphanumeric character.	The regex \w\w will match both do and d7 but not 8.
\W	It denotes any symbol.	The regex \W\W will match % and #.
{n}	It denotes the number of times the preceding character is to be repeated.	The regex Dat{3} will match Dattt.
{n,m}	n denotes the least number and m denotes the maximum number of times the preceding character is to be repeated.	The regex Data{3,6} will match Dataaa and Dataaaa but not Dataa.
\s	It denotes any white space character including 'space' and 'tab'	The regex Data\ssecurity will match Data security
\S	It denotes any non-white space character.	The regex \S\S will match &H and t8 but not 5
\D	It denotes any non-digit character.	The regex \D\D will match to and t# but not g7.

Let's consider a few examples of personal data regexes and decode it.

American Express card numbers:

RegEx: ^3[4,7][0,9]{13}*

^3 - The string starts with 3.

^3[4,7] - The string starts with 34 or 37

[0,9]{13}* - The last 13 characters are from 0 to 9

Examples: 372418640982660 and 345613290542766

Email Address:

RegEx: ^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$

^[\w - Starts with alphanumeric content.

^[\w-\.] - Starts off with alphanumeric content, '-', and '.' usage.

^[\w-\.]+ - The string can include as many characters as needed that satisfies the above condition.

@([\w-]+\.)+ - In the string @ precedes alphanumeric content, '-', and '.' usage.

[\w-]{2,4}$ - The string ends with 2 to 4 alphanumeric content.

Examples: dave.cs@xyz.com and robert52@dst2.in

Social Security Number (SSN)

RegEx: ^\d{3}-\d{2}-\d{4}*

^\d{3} - The string starts with any three whole numbers

^\d{3}-\ - The three whole numbers at the beginning of the string precedes '-'.

\d{2} - Denotes any two whole numbers

\d{4} - The string ends with any four whole numbers.

Examples: 123-45-6789 and 353-09-0007

Note: This document is intended to give the users an introduction to constructing regular expressions. In case you require any additional assistance, feel free to raise a request with our support team, and we will get back to you immediately.

Additional resources

- Regexlib is a library of most commonly used regular expressions.
- Regular-expressions.info provides a wide range of in-depth information on how-to construct a regular expression.
- Regex101 is a online regular expressions tester and debugger.