



## **Senf: the mustardy sensitive number finder**

Senf is a portable tool for finding sensitive numbers. Use this tool to identify files on your system that may have Social Security Numbers (SSNs) or Credit Card Numbers (CCNs). The latest version can always be found at the Senf site.

### **Warning! [About Senf]**

#### **Warning: Do not have a false sense of security after running this program**

It is important to understand what Senf is, and what Senf is not

### **What Senf is**

- A program written in Java by humans
- A program which can quickly and conveniently point the operator of a computer to files which contain strings of text which resemble SSNs or CCNs
- A program which tries to find files with large quantities of SSNs/CCNs; you can tell it to find a single occurrence of a pattern, but that will yield many false positives.

### **What Senf is not**

- An infallible oracle which will detect all SSNs/CCNs and only SSNs/CCNs.
- A prophet which will enlighten a computer operator as to the exact location of an SSN/CCN within a file (although the GUI version is somewhat sibylline).
- Designed to detect sensitive data in encoded/encrypted binary files. In fact, it skips many file extensions by default.

### **What all this means**

- There will be false positives
- There will be false negatives

### **Which reduces to (even though it's longer)!**

This tool is not to be regarded as the end-all in your effort to ensure your computer is free of SSN/CCN records. It simply will report to you files that contain numbers that could pose a security threat. Remember, it looks for strings of numbers -- and the typical computer has lots of these.

### **Requirements**

#### **Java 1.5 JRE**

No matter what system you're running on, you need the Java 1.5 runtime (or greater); you do not need the whole Java 1.5 SDK, which includes the runtime.

## System path

We assume that the Java interpreter is in your environment path (meaning, no matter where we try to run java from, it will run). The JRE installer should modify your system path to include the java interpreter.

If you get a strange error message saying something along the lines of "If you see this, Senf did not run!" then chances are your path is not set up correctly. Unfortunately, the solution to this is beyond the scope of this text.

---

## Installation

Once the JRE is installed, all you need to do is copy senf.jar and the seeds folder to some folder on the computer that is going to run the scan. You might also want to copy the configuration files to the same folder.

---

## Running

### Brief Note

On some Operating Systems (typically Windows and Mac OS X), simply double-clicking on the senf.jar file will launch the program automatically. If this works, you can skip the rest of this section.

### Windows

- Open a command prompt
- Navigate to the folder in which Senf is installed.
- Run `java -jar senf.jar` (with optional arguments)

### Linux and Mac OS X

- Open a command shell
- Navigate to the folder in which Senf is installed.
- Run `java -jar senf.jar` (with optional arguments)

## Using Senf

### Usage: `senf [OPTIONS]`

| Option                            | Default       | Effect  |
|-----------------------------------|---------------|---|
| <code>-h</code>                   | n/a           | display the pretty help screen  |
| <code>-q</code>                   | off           | quiet mode (display no output)  |
| <code>-v</code>                   | off           | verbose mode (display everything)   |
| <code>-e</code>                   | off           | print error messages to the screen  |
| <code>-f &lt;filesize&gt;</code>  | infinite      | Set the max file size to scan; end size (no spaces) with 'g' for gigs, 'm' for megs, 'k' for kilobytes, and nothing for bytes |
| <code>-m &lt;number&gt;</code>    | 15            | Set minimum number of times to match a CCN/SSN pattern before reporting a file  |
| <code>-p &lt;scan path&gt;</code> | working dir   | Set the path to start scanning from   |
| <code>-l &lt;yyyyMMdd&gt;</code>  | off           | Set modified-date check; files last modified before this date are skipped   |
| <code>-o &lt;log file&gt;</code>  | senf_DATE.txt | Set the name of the file (including path, if you like) where log information will be saved                                    |

|     |     |   |
|-----|-----|---|
| -ac | off | Append configuration information to the end of the output log |
| -g  | off | Hide the GUI  |
| -z  | off | Read ZIP files  |
| -as | off | Auto-start scanning (ignored when -g is specified)            |

By default, Senf only prints to the screen files which are matched -- not all output is shown.

## Examples

- To search all files in your home directory in Linux/Mac OS X
  - `java -jar senf.jar -p ~/`
- To search all files in your home directory in Windows XP
  - `java -jar senf.jar -p "C:\Documents and Settings\<yourname>"`
- To scan only files <= 100MB, ensure that each one has at least 12 matches before marking it as possible, display error messages, and start in a folder called C:\mustard\gruga
  - `java -jar senf.jar -f 100m -m 12 -e -p "C:\mustard\gruga"`

Also, note that this program may take a while to complete; again, by default, the only things it prints to the screen are possible matches (ie no errors), so it may look like it's frozen, not printing anything for a while, but it's (probably) not.

As of the Sasuke.188 release, Senf provides a GUI for ease of use. The GUI offers a results viewer to help the user quickly identify what was flagged by Senf as being sensitive. Results appear in the central pane of the Senf window as they are found; if an entry is clicked on, the Senf Analyzer will pop up, showing the applicable matches in the file.

---

## Configuration files

### Extensions

Senf includes an extensions blacklist file named `senf_extensions.conf`. This file contains a list of extensions to avoid scanning. Image files, binary executables, and compressed data are examples of files you may not want to scan, because they'll only yield false positive matches.

To avoid these files, you can add an entry (one per line!) to the blacklist. The extensions are interpreted as part of a regular expression which looks something like: `/(zip|jpe?g|exe)$/`

If any file matches the extensions list, it will not be scanned.

Entries are case insensitive.

### Blacklist

The general purpose blacklist file `senf_blacklist.conf` lets you specify specific files, or whole paths, that you do not want to scan.

There can be two types of entries in the blacklist file:

#### Explicit match

Explicit matches tell Senf to ignore only files which match exactly that path. For example, `c:\windows\system32\msvcrt.dll` will skip only the `msvcrt.dll` file.

## Regular expression

Entries in the blacklist file, if prepended with a '#' character, are interpreted as regular expression matches. Regular expression matches can be used to ignore whole paths, or even classes of paths.

Examples are:

- Ignore any file in or under the windows directory (note the double-backslashes)
  - #c:\\windows\\
- Ignore any dll file
  - #\\.dll\$
- Ignore any files in or under the lib directory
  - #.\*lib/.\*
- Functionally identical to the expression above
  - #lib/

Regular expression matches are case insensitive.

## Whitelist

The general purpose whitelist file `senf_whitelist.conf` overrides any settings in the general purpose blacklist, but not the extensions blacklist. Its syntax is identical to the blacklist configuration file.

---

## Algorithms

Senf looks for certain patterns to reduce false positives. Those patterns are described here. These patterns cannot be used to find every conceivable incarnation of the numbers Senf searches for. However, if you have suggestions for improving the algorithms (and, better, known false negatives to back up your suggestions) please let us know.

## Important

As of Sasuke.188, Senf is modular: that is, the SSN/CCN scanning capabilities have been moved out of the core Senf engine and moved to extension files placed in the seeds directory. This allows for future plugins.

## Credit card numbers

### Formats

There are a number of valid credit card formats. Senf supports only the 16 digit formats. This includes Mastercard, some (but evidently not all) VISA, and Discover. It does NOT include, for example, American Express.

### Separators

Credit cards numbers may be one long string of numbers (nnnnnnnnnnnnnnnn), or may be separated into groups of four digits (nnnn-nnnn-nnnn-nnnn). There are, of course, as many ways to delimit groups of digits as can be imagined; Senf only counts matches that use either no separator, or only one of:

- dash ("-")
- space (" ")
- dot (".")
- pipe ("|")

**Luhn check**

Credit cards must pass a Luhn mod 10 check to be considered valid.

**Social Security Numbers****Formats and separators**

Socials are detected in both single string (nnnnnnnnn) and grouped (nnn-nn-nnnn) formats; permitted separators are the same as credit card numbers.

**Validity checking**

Socials are verified against their area (the first three digits), according to the Social Security Administration's current list of valid high groups. In addition, group and serial numbers may not be all zeroes.