

LuaCensor: A package for redacting sensitive information

Elijah Z Granet*

18 February 2022

Contents

1	Very Quick Guide	2
1.1	Purpose	2
1.2	Usage	2
1.3	Requirements	2
1.4	Warning	2
1.5	Demonstration	3
2	More detailed information	4
2.1	Purpose	4
2.2	The censoring mechanism	4
2.3	The accessibility feature	4
2.4	Bugs and development	5
2.5	Licensing	5
2.6	Some useful advice	5
3	Implementation	6
3.1	Dependencies	6
3.2	Fonts	7
3.3	Eliminate pesky environments	8
3.4	The <code>cnsr</code> command	9
3.5	The Lua Magic	11

*e-mail: ezg21@cantab.ac.uk

1 Very Quick Guide

1.1 Purpose

This package redacts sensitive information using Lua, and adds accessibility support.

1.2 Usage

The package is called with:

```
\usepackage{luacensor}
```

Sensitive information is enclosed within the command:

```
\cnsr{John Smith}
```

When the outputted document is intended for authorised readers who are supposed to see sensitive information, no further action is needed. When the outputted document is for general audiences, who are *not* supposed to see sensitive information, add the following line to the preamble:

```
\cnsrtrue
```

This activates the censoring globally.

1.3 Requirements

This package will **only** work in Lua \LaTeX . The package works out of the box with a standard \TeX distribution, but ideally, I would strongly suggest installing the free (*libre & gratis*) ‘Redacted’ font from Google Fonts, which gives a quite aesthetically pleasing black bar effect.¹

1.4 Warning

The package is completely effective at censoring text formatted with normal \LaTeX commands, by which I mean that it is impossible for someone to ascertain the original text (or even its precise length) from the outputted \pdf when the `\cnsrtrue` option has been activated.

However, math mode is used, it will censor numbers, but not operators or \TeX (as opposed to Unicode) operators. This is probably fine for most instances, but unacceptable where security is of the highest priority, and I would not really recommend using this package to censor highly secret formulæ; the censor package in your \TeX distribution will do a better job of that. The package is set to completely disappear the output (as opposed to black bar over) of the

¹Available at: <https://fonts.google.com/specimen/Redacted>.

math, align, equation, tabular, and a few other environments, as disappearing these environments proved more secure than the piecemeal blacking out I saw. It is probable that there are packages and macros that will break the `cnsr` macro, and therefore, care should be taken to always examine output before public distribution.

1.5 Demonstration

```
%In the preamble: \usepackage{luacensor}
\begin{quote}
\cnsrtrue\footnotesize Whereas recognition of the \cnsr{inherent dignity
↳ and of the equal and inalienable rights of all members of the human
↳ family} is the foundation of freedom, justice and peace in the world,

\cnsr{Whereas disregard and contempt for human rights have resulted in
↳ barbarous acts which have outraged the conscience of mankind, and the
↳ advent of a world in which human beings shall enjoy freedom of speech
↳ and belief and freedom from fear and want has been proclaimed as the
↳ highest aspiration of the common people,}
```

Whereas recognition of the [REDACTED] is the foundation of freedom, justice and peace in the world,

2 More detailed information

2.1 Purpose

This package is a relatively lightweight and aesthetically pleasing censorship solution which includes accessibility features to allow screen readers to be aware that content has been redacted.

2.2 The censoring mechanism

The package uses Lua's `toks` filter to replace all `UTF8` characters with a single glyph (• in the case of `Redacted`, and a Unicode black rectangle in the fallback `TeX` default font `Source Sans Pro`). In both font options, these combine visually into a single line (though this can be deconstructed in a text editor).

However, while changing all characters into a single character is effective in *most* cases, this alone would not be sufficiently secure for reliable usage. This is because knowing the length of a censored name could be combined with other information in, for example, a Family Court judgment, to allow for what lawyers call 'jigsaw identification' (*eg*, where there is only one person with an eight letter surname who meets the other details given in the judgment).

Therefore, the package adds an extra layer of security by randomly changing the length of strings during the censorship phase; censored strings can thus be either longer or shorter by a few characters. This means that while the area of the blacked out content will be *approximately* similar to the length of the uncensored string (which means wireframing more or less works), it cannot be used to reverse engineer information about the censored content.

2.3 The accessibility feature

One concern about document redaction is ensuring that visually impaired readers of your document, who use screen reading software to listen to your text, may encounter problems with censored content. If the screen reading software skips over the censored text altogether, it will be a very confusing jump for the visually impaired user. If the screen reading software reads the replacement characters, it will be very annoying for the visually impaired reader to hear, in a censored paragraph, the same character being read out *ad nauseam* (*eg*, 'Asterisk, asterisk, asterisk...').

To overcome this limitation, the package uses the `accsup` package to add an 'actual text' feature which will lead screen readers (and utilities like `pdftotext`) to replace the string of replacement characters with the two words 'TEXT REDACTED'. This also will be encountered by naïve users who try to outdo the package by copying and pasting the black blocks from Adobe™ Acrobat or Reader. (However, because other PDF readers, like Apple's Preview, do not implement accessibility features, this is **not** an additional security feature and is not on its own sufficient to work for redaction; if it were otherwise, the rest of the package would be unnecessary)

2.4 Bugs and development

All bugs, feature requests, or other technical points should be submitted to the package's official Github page.²

2.5 Licensing

The software is free and open-source software licensed under the Latex Public Project Licence.³

2.6 Some useful advice

This package is really good at some things, but if you find it breaks down on censoring complex \LaTeX code, the existing `sensor` package on CTAN is excellent (albeit less good with accessibility), and works with non-Lua versions of \TeX . Incidentally, you can use both this package and `sensor` in the same file without trouble; this (*not* a penchant for annoying tech-speak) is why the main command in this package is `cnsr` without vowels.

²<https://github.com/ezgranet/luacensor>

³<https://www.latex-project.org/lppl/>

3 Implementation

```
6 \ProvidesPackage{luacensor}  
7 [2022/02/17  
8 Redact sensitive information using Lua]  
9 % This work may be distributed and/or modified under the  
10 % conditions of the LaTeX Project Public License, either version 1.3  
11 % of this license or (at your option) any later version.  
12 % The latest version of this license is in  
13 % http://www.latex-project.org/lppl.txt  
14 % and version 1.3 or later is part of all distributions of LaTeX  
15 % version 2005/12/01 or later.  
16 %  
17 % This work has the LPPL maintenance status `maintained'.  
18 %  
19 % The Current Maintainer of this work is Elijah Z Granet
```

3.1 Dependencies

```
20 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
21 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
22 % DEPENDENCIES  
23 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
24 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
25 \RequirePackage{luacode}  
26 \RequirePackage{environ}  
27 \RequirePackage{verbatim}  
28 % ^ for the censoring  
29 \RequirePackage{accsupp}  
30 %^for accessibility  
31 \RequirePackage{fontspec}  
32 %^for black lines  
33 %in theory, you could do  
34 %a lighter version of this  
35 %package with just asterisks  
36 %or `[TEXT-REDACTED]'  
37 %And perhaps that's better for  
38 %the environment with printing
```

3.2 Fonts

```
39 %%%%%%%%%%
40 %%%%%%%%%%
41 % FONTS
42 %%%%%%%%%%
43 %%%%%%%%%%
44 %%%%%%%%%%
45 %%%%%%%%%%
46 % redacted is prettier and free to download
47 %%%%%%%%%%
48 %%%%%%%%%%
49 % Strongly recommended
50 %%%%%%%%%%
51 %%%%%%%%%%
52 %%%%%%%%%%
53 %%%%%%%%%%
54 \IfFontExistsTF{Redacted}{%
55 \newfontface\cnsrfnt[%%%%%%%%%
56 %the scale is arbitrary, but kind of works
57 Scale=1.1,
58 %%the below declarations are to prevent warnings about shapes not being
59   ↳ available
60 ItalicFont=Redacted,%
61 BoldItalicFont=Redacted,%
62 BoldFont=Redacted,%
63 SmallCapsFont=Redacted]{Redacted}
64 \newcommand{\onething}{\cnsrfnt\ • }
65 \newcommand{\twothings}{\cnsrfnt\ • • }
66 \newcommand{\donothing}{}
67 %%%%%%%%%%
68 %The little spaces let justification happen
69 %%%%%%%%%%
70 %%%%%%%%%%
71 %%%%%%%%%%
72 %%%%%%%%%%
73 % x chosen as an arbitrary average width
74 %%%%%%%%%%
75 %%%%%%%%%%
76 }{
77 %%%%%%%%%%
78 %%%%%%%%%%
79 % This option works perfectly
80 %fine, it's just less pretty
81 %%but a good fallback because
82 % Source Sans is in TeX dists by default
```

```

83 %%%%%%%%%%%
84 %%%%%%%%%%%
85 \newfontface\cnsrfnt[Scale=1.01,%To allow for separate use of source sans in
   ↳ text
86 WordSpace=0,%To make it all one black line
87 %the below declarations are to prevent warnings about shapes not being
   ↳ available
88 ItalicFont={Source Sans Pro Black},BoldItalicFont={Source Sans Pro
   ↳ Black},BoldFont={Source Sans Pro Black},SmallCapsFont={Source Sans Pro
   ↳ Black}}{Source Sans Pro Black}
89 %%%%%%%%%%%
90 %%%%%%%%%%%
91 % Bit of unicode magic below to make the black line effect
92 %%%%%%%%%%%
93 %%%%%%%%%%%
94 \newcommand{\onething}{\cnsrfnt - }
95 \newcommand{\twothings}{\cnsrfnt - - }
96 \newcommand{\donothing}{}
97 }

```

3.3 Eliminate pesky environments

```

98 %%%%%%%%%%%
99 %%%%%%%%%%%
100 %%%%%%%%%%%
101 %%%%%%%%%%%
102 % A neat fallback for disappearing things...
103 %%%%%%%%%%%
104 %%%%%%%%%%%
105 %%%%%%%%%%%
106 %%%%%%%%%%%
107 % FULL CREDIT
108 % and FULSOME THANKS
109 % TO TEX.SE USER
110 % Werner for the code below
111 %%%%%%%%%%%
112 %%%%%%%%%%%
113 \makeatletter
114 \newcommand{\voidenvironment}[1]{%
115   \expandafter\providecommand\csname env@#1@save@endcsname{}\%
116   \expandafter\providecommand\csname env@#1@process\endcsname{}\%
117   \@ifundefined{#1}{}\{\RenewEnviron{#1}\}\}%
118 }
119 \makeatother

```



```

120 \newcommand{\hddn}[1]{%
121 \ifcnsr{}\else%
122 #1\fi}
123 \newenvironment*{hidden}{\begin{@empty}
124 }{\end{@empty}}
125 \voidenvironment{hidden}
126

```

3.4 The cnsr command

```

127
128 %%%%%%%%%%
129 %%%%%%%%%%
130 %%%%%%%%%%
131 %%%%%%%%%%
132 %%%%%%%%%%
133 %%%%%%%%%%
134 % the CENSOR COMMAND
135 %%%%%%%%%%
136
137 \newif\ifcnsr\cnsrfalse
138
139 \newcommand{\cnsr}[1]{%
140 \ifcnsr{%
141 \voidenvironment{equation*}
142 \voidenvironment{equation}
143 \voidenvironment{table}
144 \voidenvironment{table*}
145 \voidenvironment{tabular}
146 \voidenvironment{tabular*}
147 \voidenvironment{}}
148 %%%%%%%%%%
149 % I don't know how many
150 % people use TEX native accent commands
151 % in LuaTEX given that using Unicode is more
152 % people's style. But just in case, because these can lead to stray accent
153 % ↪ marks floating above censored letters.
154 %%%%%%%%%%
155 %%%%%%%%%%
156 \renewcommand{\`}[1]{}
157 \renewcommand{\'}[1]{}
158 \renewcommand{\^}[1]{}
159 \renewcommand{\"}[1]{}

```

```

160 \renewcommand{\H}[1]{}
161 \renewcommand{\~}[1]{}
162 \renewcommand{\c}[1]{}
163 \renewcommand{\k}[1]{}
164 \renewcommand{\l}[1]{}
165 \renewcommand{\=}[1]{}
166 \renewcommand{\b}[1]{}
167 \renewcommand{\.}[1]{}
168 \renewcommand{\d}[1]{}
169 \renewcommand{\r}[1]{}
170 \renewcommand{\u}[1]{}
171 \renewcommand{\v}[1]{}
172 \renewcommand{\t}[1]{}
173 \renewcommand{\o}[1]{}
174 \renewcommand{\i}[1]{}
175 %%%%%%%%%%%
176 %TEX primitives can break the
177 %code because they don't have the good manners
178 %to put their arguments in brackets
179 %No offence, of course, to the
180 %honoured Prof Knuth, who had
181 %magnificent genius reasons
182 %for coding things that way
183 %it's just that i'm an idiot
184 %and therefore need things simple
185 %%%%%%%%%%%
186 %%%%%%%%%%%
187 %%%%%%%%%%%
188 \renewcommand{\hskip}[1]{}
189 \renewcommand{\vskip}[1]{}
190 \renewcommand{\raise}[1]{}
191 \renewcommand{\lower}[1]{}
192 \renewcommand{\kern}[1]{}
193 % here we have the accsupp magic
194 % this operates by replacing the 'x's
195 % or unicode black squares as the case
196 % may be with an alt text
197 % this serves a dual purpose of both making
198 %pdftotext not break with huge strings of meaningless characters
199 %but more importantly
200 % it means screen readers don't subject
201 %, their users to the meaningless reading out of unicode black squares 50
   ↵ times in a row!
202 %%%%%%%%%%%
203 %%%%%%%%%%%
204 %%%%%%%%%%%
205 %%%%%%%%%%%

```

```

206 \BeginAccSupp{method=plain,ActualText={TEXT REDACTED}}%
207 \rndstring{#1}
208 \EndAccSupp{}%
209 }
210 \else%
211 %%%%%%%%%%%
212 %%%%%%%%%%%
213 % if the conditional is off
214 % the command does absolutely nothing
215 %%%%%%%%%%%
216 %%%%%%%%%%%
217 #1%
218 \fi}

```

3.5 The Lua Magic

```

219 %%%%%%%%%%%
220 %%%%%%%%%%%
221 % %%%%%%%%%%%
222 %%%%%%%%%%%
223 % The LUA MAGIC PART
224 %%%%%%%%%%%
225 %%%%%%%%%%%
226 %%%%%%%%%%%
227 %%%%%%%%%%%
228 \begin{luacode}

```

```

229 --fulsome thanks to TeX.SE users Henri Menke and David Carlisle, without whom
↳ none of this would be possible
230 local function rndstring()
231
232     local toks = token.scan_toks(s)
233     local on = ttrue
234     for n, t in ipairs(toks) do
235         if t.csname == "begin" or t.csname == "end" then
236             on = false
237         -- The below is necessary as TeX primitives can break the code otherwise
238         ↳ because they do not use brackets
239         end
240
241         if not(on) and t.cmdname == "right_brace" then

```

```

241         on = true
242         -- This prevents needless errors about gibberish up commands
243     end
244     if on and t.csname == "&" then
245         local letter = token.create'donothing'
246         toks[n] = letter
247
248     elseif on and t.csname == "%" then
249         local letter = token.create'donothing'
250         toks[n] = letter
251
252     elseif on and t.csname == "$" then
253         local letter = token.create'donothing'
254         toks[n] = letter
255
256     elseif on and t.csname == "#" then
257         local letter = token.create'donothing'
258         toks[n] = letter
259
260     elseif on and t.csname == "_" then
261         local letter = token.create'donothing'
262         toks[n] = letter
263
264     elseif on and t.csname == "{" then
265         local letter = token.create'donothing'
266         toks[n] = letter
267
268     elseif on and t.csname == "}" then
269         local letter = token.create'donothing'
270         toks[n] = letter
271
272     elseif on and t.csname == "~" then
273         local letter = token.create'donothing'
274         toks[n] = letter
275
276     elseif on and t.csname == "^" then
277         local letter = token.create'donothing'
278         toks[n] = letter
279     elseif on and t.cmdname == "letter" then
280         -- The below is the randomness part of this, which I admit is fairly
281         -- arbitrary,
282         --but will more often artificially shorten
283         --strings than lengthen them, as testing
284         --found if lengthening was too frequent, it
285         --led to really unsightly long strings.
286
287         local f = math.random (1,20)

```

```

287         if f == 1 then
288             local letter = token.create'donothing'
289             toks[n] = letter
290
291             elseif f == 2 then
292                 local letter = token.create'donothing'
293                 toks[n] = letter
294             elseif f == 3 then
295                 local letter = token.create'donothing'   toks[n] = letter
296             elseif f == 4 then
297                 local letter = token.create'twothings'
298                 toks[n] = letter
299             elseif f == 5 then
300                 local letter = token.create'donothing'   toks[n] = letter
301
302             else
303                 local letter = token.create'onething'
304                 toks[n] = letter
305             end
306             elseif
307                 on and t.cmdname == "spacer" then
308                 local f = math.random (1,20)
309                 if f == 2 then
310                     local letter = token.create'donothing'
311                     toks[n] = letter
312                     elseif f == 3 then
313
314                         local letter = token.create'donothing'   toks[n] = letter
315                     elseif f == 4 then
316                         local letter = token.create'donothing'
317                         toks[n] = letter
318                     elseif f == 5 then
319
320
321                         local letter = token.create'twothings'   toks[n] = letter
322                     elseif f == 6 then
323
324
325                         local letter = token.create'donothing'   toks[n] = letter
326                         elseif f == 7 then
327                             local letter = token.create'donothing'   toks[n] = letter
328
329
330                     else
331                         local letter = token.create'onething'
332                         toks[n] = letter
333

```

```

334         end
335
336     elseif on and t.cmdname == "other_char" then
337         local f = math.random (1,20)
338
339     if f == 2 then local letter = token.create'donothing'
340         toks[n] = letter
341
342     elseif f == 3 then
343         local letter = token.create'donothing' toks[n] = letter
344
345     elseif f == 4 then
346         local letter = token.create'donothing'
347         toks[n] = letter
348
349     elseif f == 5 then
350         local letter = token.create'twothings'
351         toks[n] = letter
352
353     elseif f == 6 then
354         local letter = token.create'donothing' toks[n] = letter
355
356     elseif f == 7 then
357         local letter = token.create'donothing' toks[n] = letter
358
359     else local letter = token.create'onething'
360         toks[n] = letter
361
362         end
363     end
364     end
365     end
366     --Drop the token in and move on
367     token.put_next(toks)
368     end
369     local lft = lua.get_functions_table()
370     --make a global command
371     lft[#lft + 1] = rndstring
372     token.set_lua("rndstring", #lft, "global")

```

```

373 \end{luacode}

```