

Steg removal in corporate environment

Samuel Johnson
1731010035

A brief overview

- Corporate environment deals with confidential data/code like new operating system (Windows), proprietary software (Photoshop) code, etc.
- Strict policies can be enforced to restrict employees from bringing physical writeable media (Flash drives, CDs etc) to office premises.
- The only option a disgruntled employee is left with is to hide the data in a message and send via the corporate e-mail.

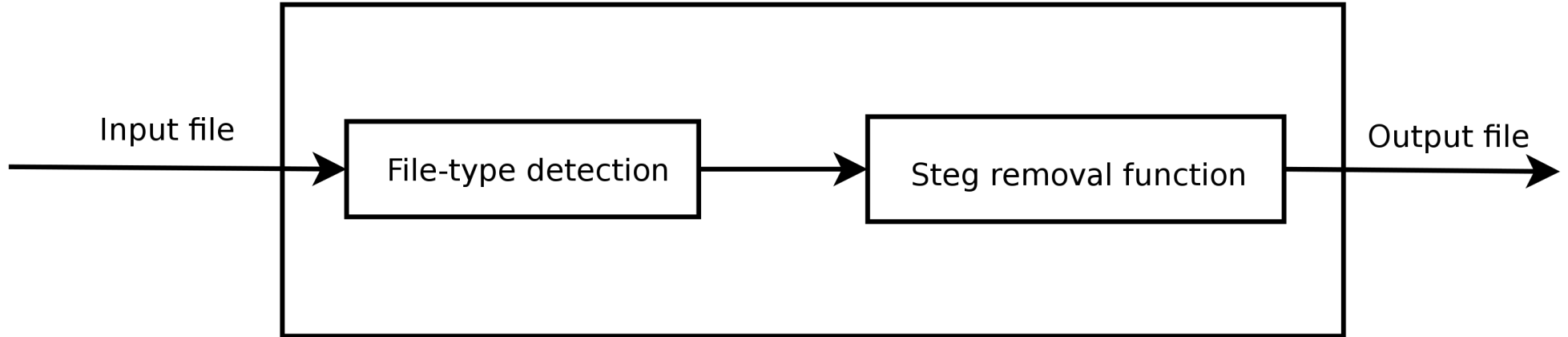
Existing systems

- Systems exist that detect certain kinds of steganography.
- However, they are largely limited by the types of steganography that they can detect.
- Newer / proprietary forms of steg which are not present in the signature database are often missed by these algorithms.

Proposed System

- The proposed system does away with the *detection* part and concentrates only on *removal* of *almost all* forms of steganography.
- The system, instead of relying on signatures, uses a transformation function that deteriorates any possible forms of steganography.
- The function is file-type specific, thus a limited number of such functions is enough to cover major forms of steganography.

Design



Modules

- Filetype detection Module

The main function of this module is to detect the type of the input file to determine the type of steg removal algorithm to apply.

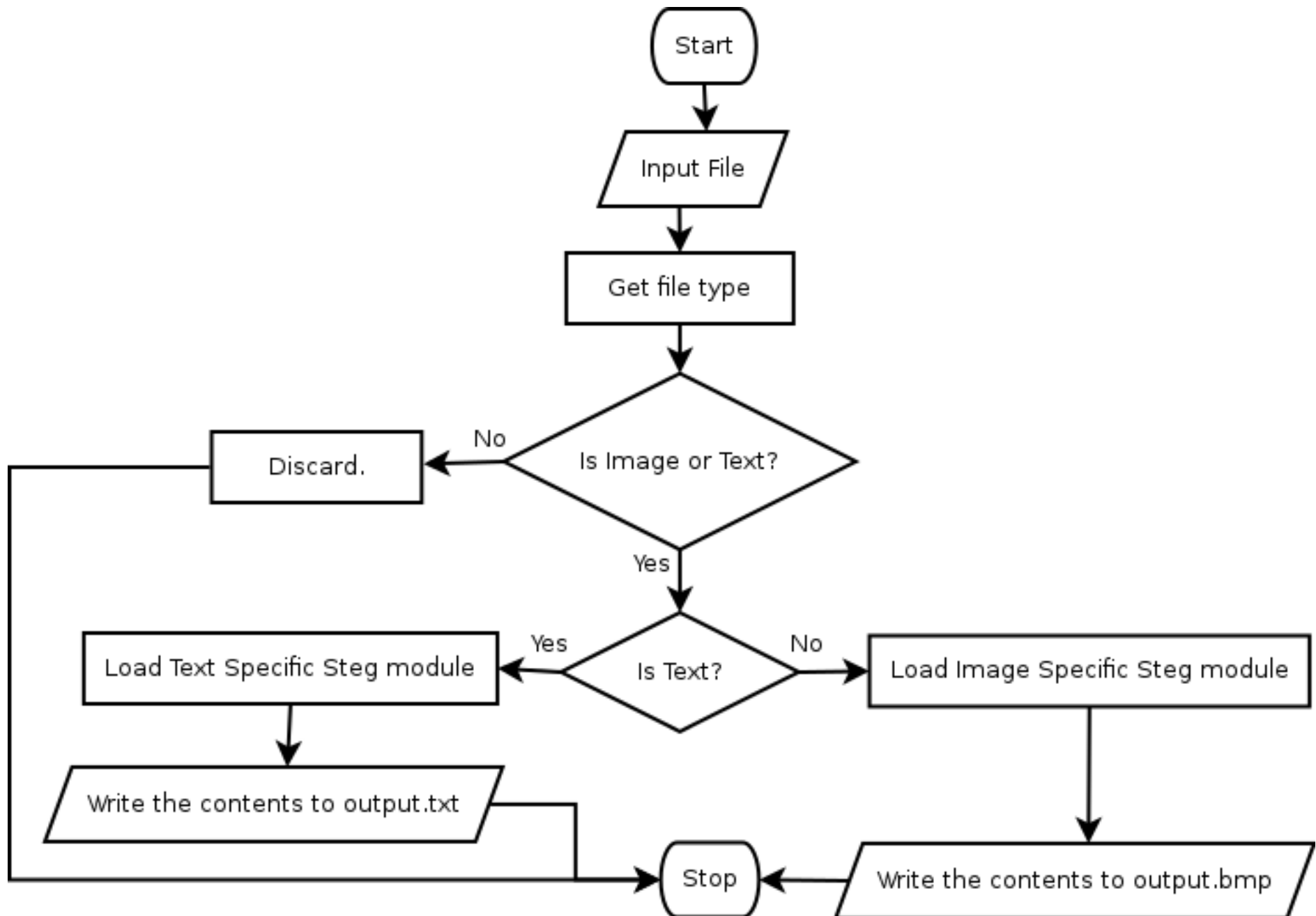
- Image Module

The main function of this module is to remove the steg content from image files.

- Text Module

The main function of this module is to remove the steg content from text files.

Flow Chart



File-type detection implementation

- The first step to steg removal in our design is to find out the file type.
- Once the type of file is found, file-type specific operations can be carried out.
- Every file type contains a unique signature in its header called the 'Magic number'.
- The file-type is detected using this magic number.

File-type detection (Sample Code)

```
const char *magic_full;  
magic_t magic_cookie;  
magic_cookie = magic_open(MAGIC_MIME);  
if (magic_cookie == NULL) {  
    printf("unable to initialize magic library\n");  
    return 1;  
}  
if (magic_load(magic_cookie, NULL) != 0) {  
    printf("cannot load magic database - %s\n",  
        magic_error(magic_cookie));  
    magic_close(magic_cookie);  
    return 1;  
}  
magic_full = magic_file(magic_cookie, actual_file);
```

Image Module (Sample Code)

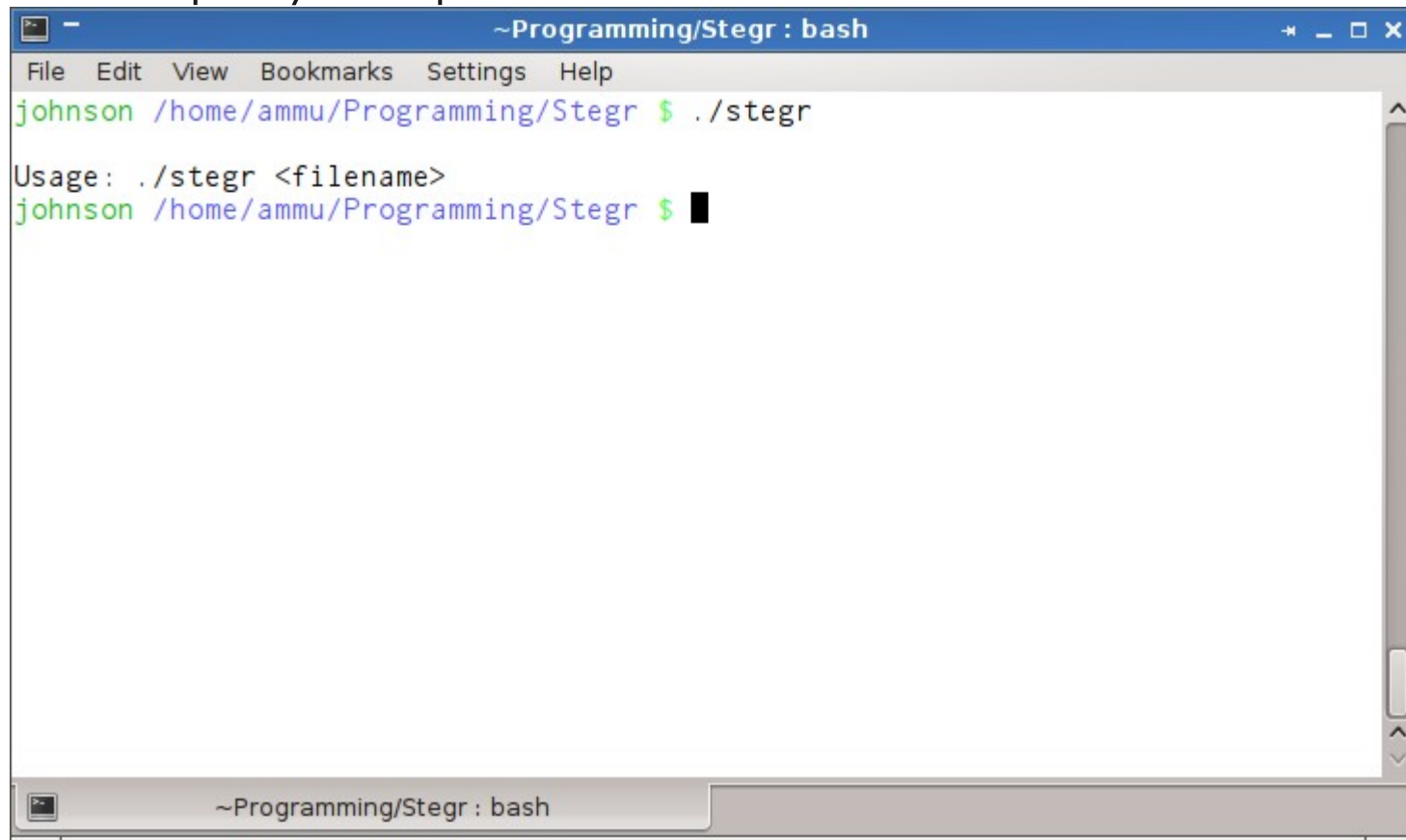
```
void imagemod(){  
    printf("Loading image specific steg module.\n");  
    MagickWandGenesis();  
    magick_wand=NewMagickWand();  
    status=MagickReadImage(magick_wand,file);  
    MagickResizeImage(magick_wand,width,height,Lancz  
osFilter,blurfactor);  
    MagickSetImageCompressionQuality(magick_wand,ima  
gequality);  
}
```

Text Module (Sample Code)

```
void textmod(){  
    printf("Loading text specific steg module.\n");  
    //Removing white space  
    while((c=fgetc(in))!=EOF){  
        if(c==' ' && pastc==' ' )  
            continue;  
        fputc(c,out);  
        pastc=c; //previous value of character saved for  
        next iteration.  
    }
```

Main Interface design

- Since the tool is to be implemented in mail serveres, it features minimallistic design without any fancy graphics and with the ability to read and write to and from standard input/output.

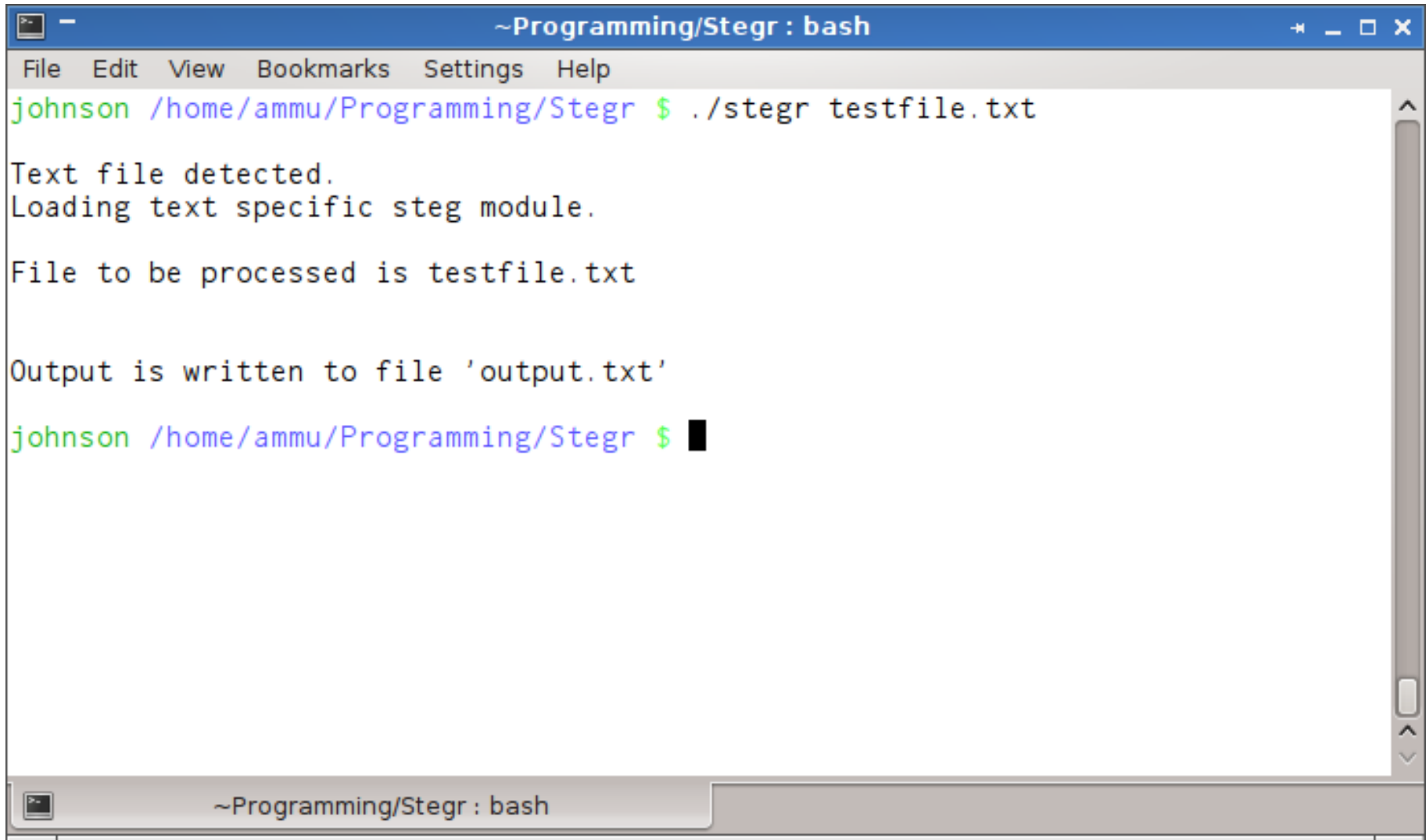


```
~Programming/Stegr : bash
File Edit View Bookmarks Settings Help
johnson /home/ammu/Programming/Stegr $ ./stegr
Usage: ./stegr <filename>
johnson /home/ammu/Programming/Stegr $ █
```

The screenshot shows a terminal window titled '~Programming/Stegr : bash'. The window has a menu bar with 'File', 'Edit', 'View', 'Bookmarks', 'Settings', and 'Help'. The prompt is 'johnson /home/ammu/Programming/Stegr \$'. The user has entered './stegr', and the output is 'Usage: ./stegr <filename>'. The prompt is now 'johnson /home/ammu/Programming/Stegr \$' with a cursor. The terminal has a scrollbar on the right and a status bar at the bottom showing '~Programming/Stegr : bash'.

Modes of Operation

- Text File



```
~Programming/Stegr : bash
File Edit View Bookmarks Settings Help
johnson /home/ammu/Programming/Stegr $ ./stegr testfile.txt

Text file detected.
Loading text specific steg module.

File to be processed is testfile.txt

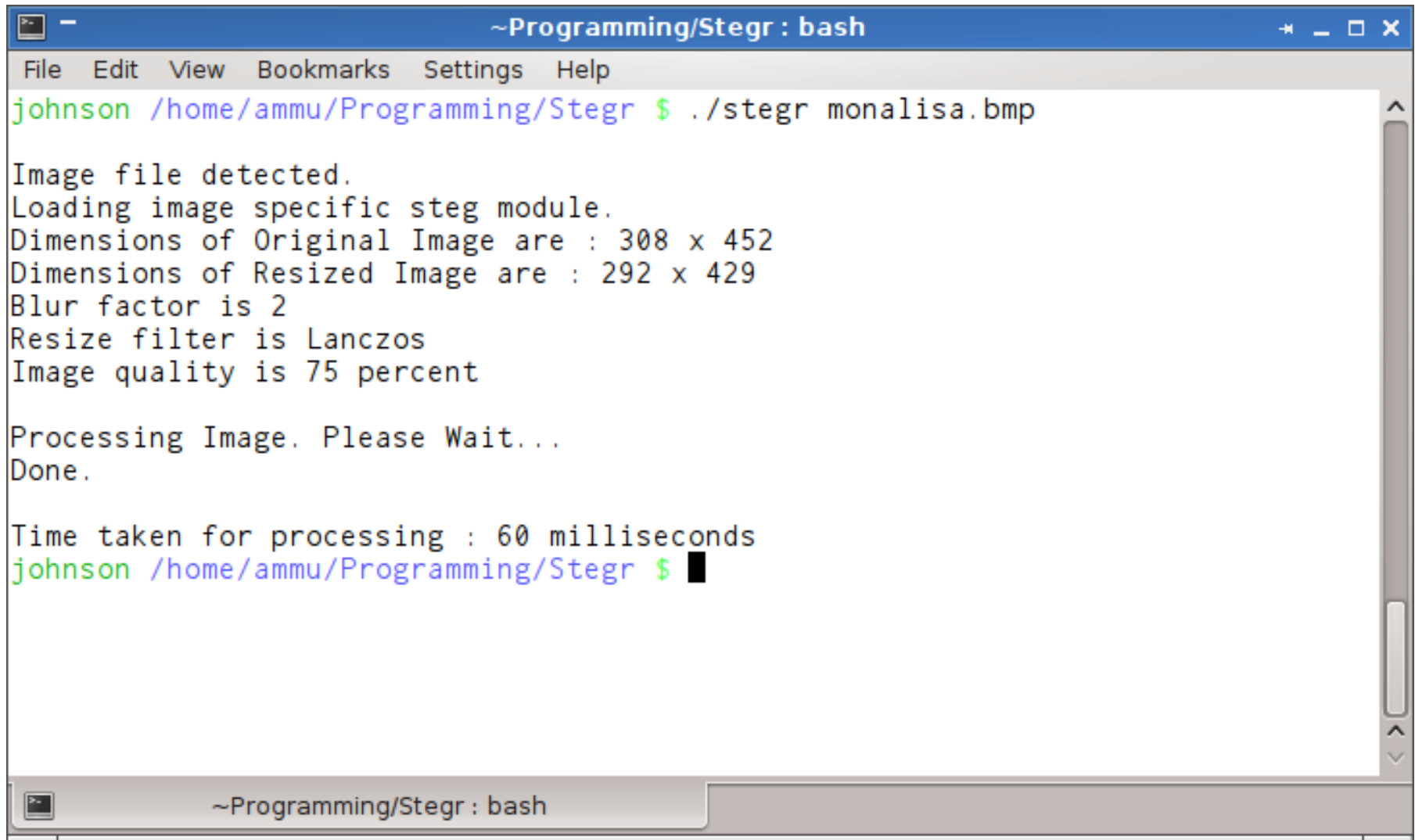
Output is written to file 'output.txt'

johnson /home/ammu/Programming/Stegr $
```

The image shows a terminal window titled '~Programming/Stegr : bash'. The window has a menu bar with 'File', 'Edit', 'View', 'Bookmarks', 'Settings', and 'Help'. The terminal content shows a user named 'johnson' in the directory '/home/ammu/Programming/Stegr' running the command './stegr testfile.txt'. The program outputs the following messages: 'Text file detected.', 'Loading text specific steg module.', 'File to be processed is testfile.txt', and 'Output is written to file 'output.txt''. The prompt returns to the user, and a black cursor is visible at the end of the line.

Modes of Operation (Continued...)

- Image File



```
~Programming/Stegr : bash
File Edit View Bookmarks Settings Help
johnson /home/ammu/Programming/Stegr $ ./stegr monalisa.bmp

Image file detected.
Loading image specific steg module.
Dimensions of Original Image are : 308 x 452
Dimensions of Resized Image are : 292 x 429
Blur factor is 2
Resize filter is Lanczos
Image quality is 75 percent

Processing Image. Please Wait...
Done.

Time taken for processing : 60 milliseconds
johnson /home/ammu/Programming/Stegr $
```

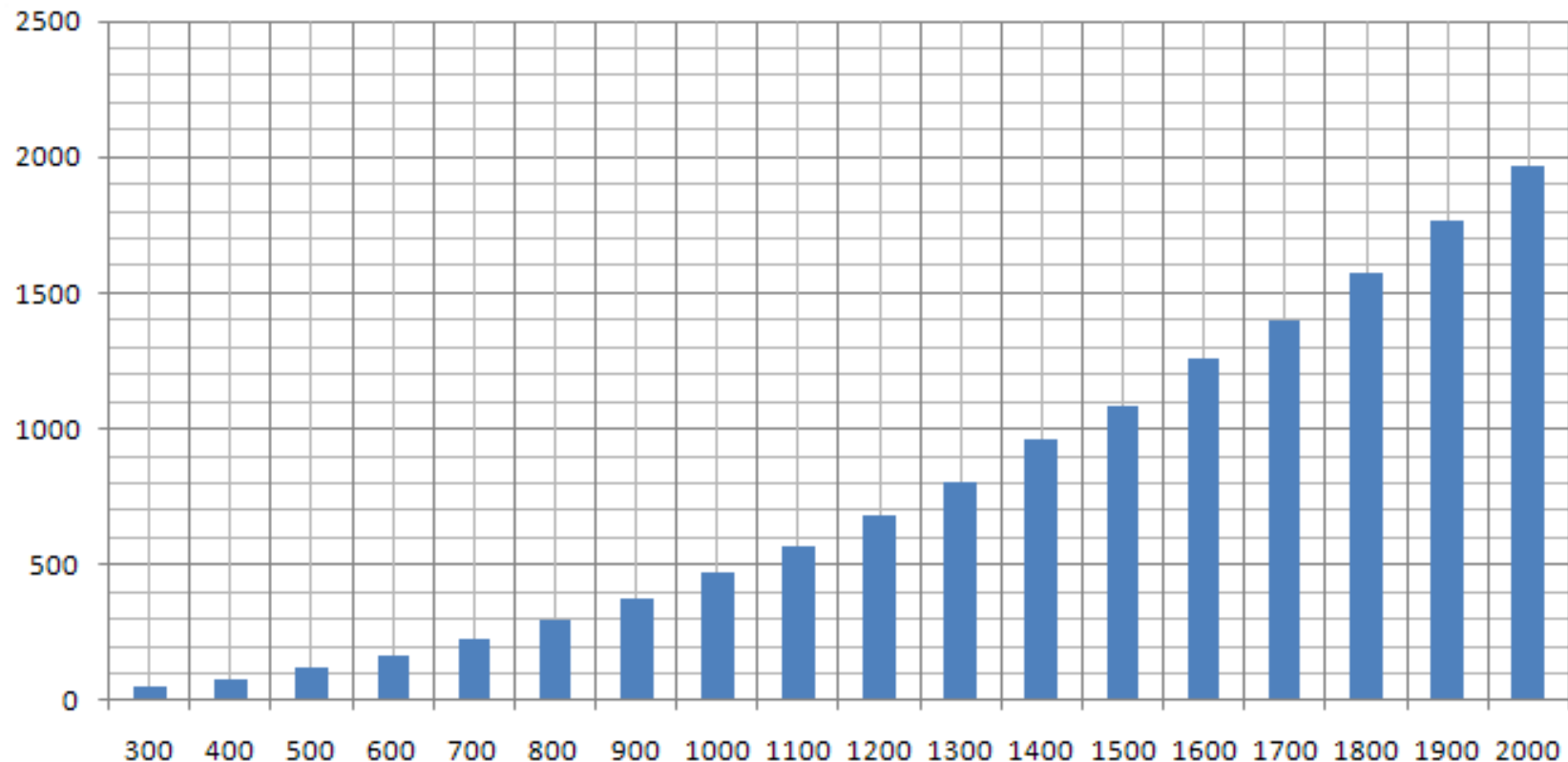
Implementation Results

- Text Processing Time

The Processing time of text is nearly constant since text manipulations are faster in Linux.

- Image Processing Time

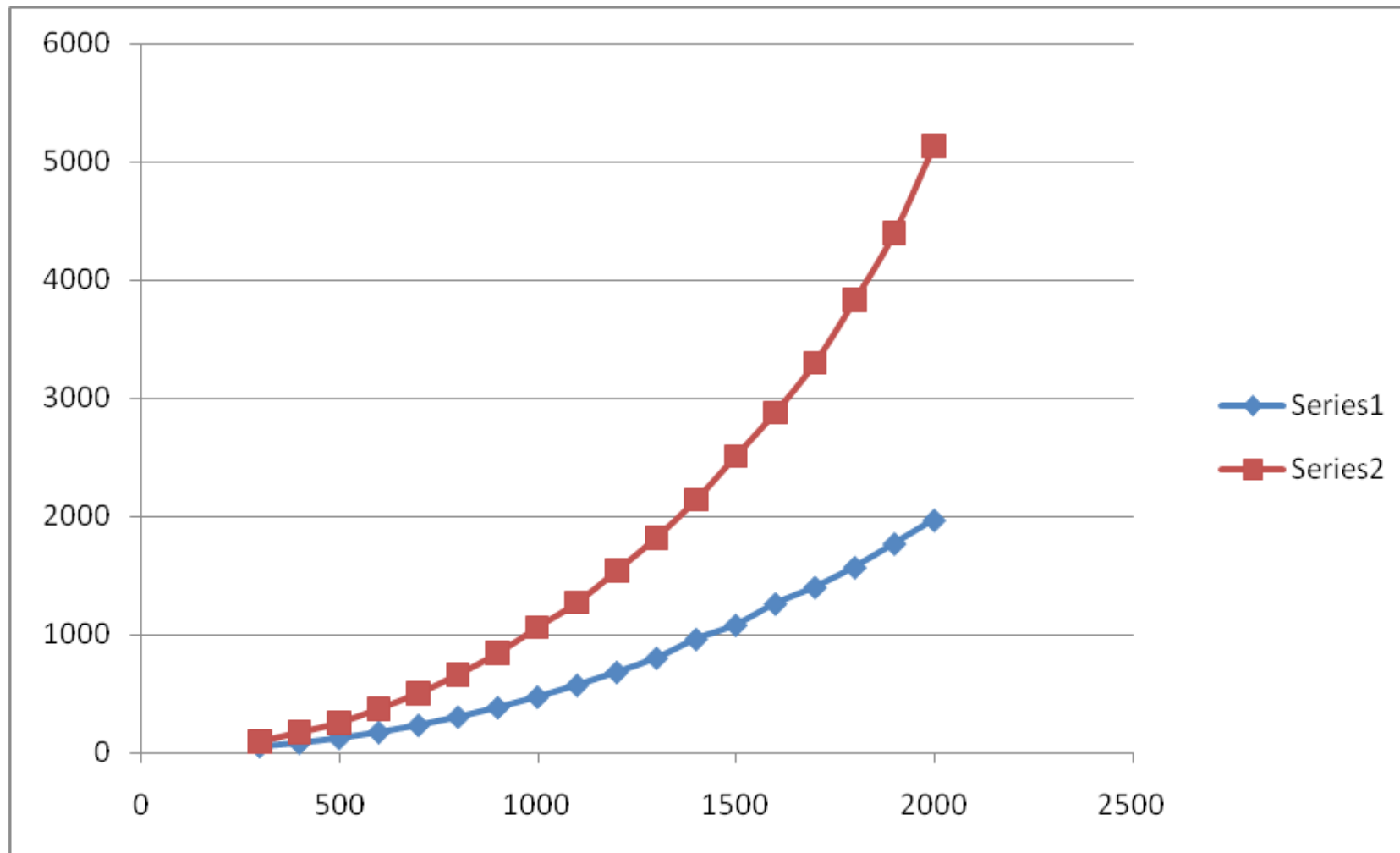
The processing time of Image however is a function of image resolution.



Implementation Results (Continued...)

- Image Processing Time

The blurring algorithm is the one which takes the most amount of time.



Conclusion and Scope for Enhancements

- It has been found that text manipulations are faster and nearly constant. Where as the processing time for image manipulation is a function of image size and is directly proportional to it.
- However it is also noted that blur factor is the one taking the most amount of time, the resize and degrade functions are relatively lightweight on the CPU.
- It can be clearly seen that there is a tradeoff between blur factor and processing time, the bigger blur factor ensures that the image is devoid of the steg content, but is a costlier operation and considerably degrades the image content. The smaller blur factor does not impact steg content much, is visually similar and is faster.

Conclusion and Scope for Enhancements (Continued...)

- The study of various parameters and operations on images and texts to effectively combat against the steg content without adversely increasing the processing time and the visual content of the image is a different study altogether.
- The tool can be further enhanced by adding support for various other filetypes like audio, video etc.
- The existing algorithm can be further extended to combat even more complex forms of steganography.

Thank you