Name:	
SBID:	
Grader:	
General Comments:	

	Max	Actual	Comments
Functionality (points earned/lost based on running your program)	**55**	**0**	
Submission guidelines are followed properly. Directory structure is as mentioned in assignment description. “make” command inside CSE-506/ works as expected. The default target, "all", makes both the user-space and kernel components. "clean" removes all temporary files.	5	0	
Cleanly compiles with gcc -Wall -Werror, and a simple test (smaller than PAGE_SIZE) successfully encrypts and decrypts; points deducted for warnings/errors. Note: If your module does not compile and load on the CS VM systems, you will lose all of the functionality points, not just this	10	0	
Several test files that are of various sizes. Your errors will probably be based on boundaries of PAGE_SIZE, cipher size multiples, zero size files, which will be tested. File sizes not at a particular boundary like PAGE_SIZE should be correctly handled.	10	0	
Rejects files that were encrypted with a different key, or when an incorrect key is specified. Returns appropriate error messages.	4	0	
Handles invalid input and reports them back to user-space. User-space informs the user of the error. There are many bad types of input, you need to check for them all. Examples: invalid filenames, files that are inaccessible, encrypt and decrypt both specified. See assignment description for more of these.	10	0	
Obeys permissions on input files. Output files have matching permissions. Other users can not access the file at any point in time.	4	0	
Partial output files are removed.	4	0	
Input and output are different, and they are regular files.	4	0	
Grader's discretion for other "broken" functionality	4	0	

Code Inspection (points earned/lost based on examination of your source)	**45**	**0**	
All user-space parameters correctly validated	5	0	
The data stored in the preamble to validate files does not compromise security. Does not, for example, store the key with any sort of reversible obfuscation.	5		
All allocated resources are correctly deallocated, even under error conditions. Unallocated resources are not deallocated.	5	0	

Code clarity, efficiency, and style. This is a rather all encompassing category, but essentially anything "bad" you do that doesn't fit elsewhere fits here. For example, wasting kernel memory; unclean system call addition code; difficult to read code (e.g., excessive gotos as control structures), "proper" indentation (you don't need to follow the kernel style of 8 space tabs, but your code must be reasonably indented), directly calling system calls from your kernel would lose you points, as would other confusions of the user and kernel address space. In user-space do you use suitable library calls for common functions (e.g., getopt for parsing command line arguments). In kernel-space do you use the appropriate functions rather than inappropriate ones or copying code? Does checkpatch.pl report any problems?	10	0	
User level code clarity and quality.	2	0	
Minimal kernel configuration developed (excluding kernel-hacking section) config options. 5pts if <1000 config options; 4pts if <1200; 3pts if <1600; 2pts if <2200; 1pt if <3000; 0 if >=3000 config options.	5	0	
Includes at least 10 useful and properly commented shell scripts that each test only one thing.	8	0	
README: Is named README. Describe all design decisions. How do you select the key size? How do you handle files that are not a multiple of the cipher block encoding size? What do you store in the preamble?	5	0	

Extra Credit	**25**	**0**	
A: An IV is used for encryption and decryption, and multiple ciphers are handled. Two identical pages in the same file are different. Two identical files in are different with IVs.	4	0	
B: Supports all ciphers supported by the CryptoAPI, and rejects invalid cipher names. Files are encrypted and decrypted properly. Padding is performed properly regardless of the cipher's block size.	6	0	
C: Multiple encryption sizes/units, key sizes.	5	0	
D: The 5 smallest working kernel config files.	5	0	
E: Extra credit at grader's discretion, for any particularly clever solutions/enhancements (up to 5 pts)	5	0	

General Demerits	**0**	**0**	
Followed GIT submission guidlines improperly (e.g., no missing/extra files committed, branches)	0	0	
Submission on time: deduct 1 point for every late hour (time rounded up in units of one hour)	0	0	
Seemlingly "random" kernel oopses that are not triggered by anything specific will cost you points. These can indicate races, memory corrpptions, etc. In general oopses will be counted under their respective category if they are easily reproducible.	0	0	

Total Grade (out of 100)	**100**	**0**	
Total Extra Credit (NOT counted as part of the total above)	**25**	**0**	