What is claimed is :

1. A method for protecting software from piracy, the method comprising:

receiving multiple binary files, each binary file comprising code and data, wherein the code of each binary file includes a reference to the data of that binary file and the reference requires a fixed distance between the code and the data;

modifying the multiple binary files such that the data of each binary file may be located at an arbitrary distance in memory from the code of that binary file;

encrypting the code of each binary file; and

receiving, from a computing device, a request for a decryption key, wherein the computing device includes a hardware enclave, wherein the hardware enclave comprises a defined region of memory of the computing device configured to protect the hardware enclave from instructions residing outside the hardware enclave, wherein the encrypted code of each binary file is stored in the hardware enclave, and wherein the data of each binary file is stored outside of the hardware enclave.

2. The method of claim 1 further comprising: providing, to the computing device, the decryption key.

3. The method of claim 2 further comprising: authenticating, before providing the decryption key, a processor signature, wherein a processor of the computing device generated the processor signature and the request for the decryption key includes the processor signature.

4. The method of claim 3 further comprising: validating, before providing the decryption key, a hash, wherein the request for the decryption key includes the hash.

5. The method of claim 1, wherein modifying the multiple binary files includes modifying data references in the code of each binary file that require that the data be placed a fixed distance in memory from the code.

6. The method of claim 5, wherein modifying the multiple binary files is done without access to source code or debug symbols for the multiple binary files.

7. The method of claim 1 further comprising : adding a separation header to each of the multiple binary files, wherein the separation header indicates that the data of each of the multiple binary files may be placed at the arbitrary distance in the memory from the code of that binary file.

8. A system for facilitating protecting a software program from piracy, comprising:

one or more processors;

system memory in electronic communication with the one or more processors; and

instructions stored in the system memory, the instructions being executable by the one or more processors to:

receive multiple binary files, each binary file comprising code and data, wherein the code of each binary file includes a reference to the data of that binary file, and wherein the reference is designed such that the data must be stored at a fixed, predetermined distance in memory from the code;

modify the multiple binary files such that the data of each binary file may be located at an arbitrary distance in the memory from the code of that binary file;

encrypt the code of each binary file but not the data of each binary file; and

receive, from a computing device, a request for a decryption key, wherein the computing device includes a hardware enclave, wherein the encrypted code of each binary file is stored in the hardware enclave, wherein the data of each binary file is stored outside of the hardware

enclave, wherein the hardware enclave includes instructions to mark the

hardware enclave as non-readable before executing the code, and wherein the hardware enclave comprises a defined region of memory of the computing device configured with confidentiality and integrity protection from other instructions residing outside of the hardware enclave.

9. The system of claim 8, the instructions further being executable by the one or more processors to: provide, to the computing device, the decryption key.

10. The system of claim 9, the instructions further being executable by the one or more processors to: authenticate, before providing the decryption key, a processor signature, wherein a processor of the computing device generated the processor signature and the request for the decryption key includes the processor signature.

11. The system of claim 10, the instructions further being executable by the one or more processors to: validate, before providing the decryption key, a hash, wherein the request for the decryption key includes the hash and validating the hash comprises comparing the hash to a verified hash value.

12. The system of claim 8, wherein modifying the multiple binary files includes identifying data references in the code of each binary files and modifying the data references .

13. The system of claim 12, wherein modifying the multiple binary files does not require access to source code or debug symbols for the multiple binary files.

14. The system of claim 8, the instructions further being executable by the one or more processors to add a separation header to each of the multiple binary files, wherein the separation header indicates that the data of each of the multiple binary files may be placed at the arbitrary distance in the memory from the code of that binary file and that the code of that binary file should be placed in the hardware enclave.

15. A non-transitory computer-readable medium comprising instructions that are executable by one or more processors to cause a computing system to:

receive a request to initiate an application, the application comprising one or more files with executable code and data, wherein the executable code is encrypted;

load the executable code into a hardware enclave on the computing system;

load the data into a location in memory on the computing system, wherein the location is outside the hardware enclave and is not a predetermined distance in the memory on the computing system from the executable code, wherein the hardware enclave comprises a defined region of the memory and only software stored in the hardware enclave can read or save contents of the hardware enclave;

send a request for a decryption key to an authentication server;

decrypt the executable code using the decryption key, wherein the decryption key is received by the computing system after sending the request for the decryption key; and

mark the hardware enclave as non-readable.

16. The non-transitory computer-readable medium of claim 15, wherein the executable code comprises two or more sections of executable code and the two or more sections of executable code are loaded in a continuous range in the hardware enclave.

17. The non-transitory computer-readable medium of claim 15, further comprising additional instructions that are executable by the one or more processors to cause the computing system to: load startup code into the hardware enclave.

18.. The non-transitory computer-readable medium of claim 17, further comprising additional instructions that are executable by the one or more processors to cause the computing system to: measure a hash of the startup code and the executable code.

19. The non-transitory computer-readable medium of claim 18, further comprising additional instructions that are executable by the one or more processors to cause the computing system to: sign the hash with a signature of the one or more processors.

20. The non-transitory computer-readable medium of claim 19, wherein the request for the decryption key includes the signature and the hash.