CLAIMS:

1. A boot image randomisation method for an Android RunTime (ART) virtual machine during a system boot comprising:

creating an image space in an ART heap; wherein the image space is used to load a boot image;

deleting the boot image in a cache path;

relocating the boot image according to a new patch delta in response to the boot image is not available in the cache path by removing all applications' OAT caches, selecting a patch delta randomly, patching and writing the boot image to a path for caching, executing an image space initialization and loading the boot image to the image space.

2. The boot image randomisation method according to claim 1, wherein after the step of relocating the boot image according to a new patch delta, further comprising:

creating a Class Linker according to the ART heap such that when a system class is going to be linked based on the Class linker, an image header will be obtained from the image space in the heap the patched address of boot image will be used to find the system class.

3. The boot image randomisation method according to claim 1 or 2, wherein the boot image contains a boot.oat file and wherein the step of relocating the boot image according to a new patch delta further comprising:

selecting gaps randomly in a text section of the boot.oat file; and

patching the boot.oat file according to the selected gaps.

4. A boot image randomisation method for an Android RunTime (ART) virtual machine during a system boot comprising:

creating an image space in an ART heap, wherein the image space is used to load a boot image;

checking whether the boot image is cached in a cache path;

loading the cached boot image in the image space in response to the boot image is available in the cache path;

relocating the boot image according to a new patch delta in response to the boot image is not available in the cache path by removing all applications' OAT caches, selecting a patch delta randomly, patching and writing the boot image to a path for caching, executing an image space initialization and loading the boot image to the image space;

creating a Class Linker according to the ART heap; wherein the step of loading the cached boot image in the image space in response to the boot image is available in the cache path comprises:

reading an image header from the cached boot image, in which the patch delta for determine loading address is stored;

modifying the patch delta randomly to load the boot image in a new address.

5. The boot image randomisation method according to claim 4, further comprising:

amending the Class Linker according to the modified patch delta.

6. The boot image randomisation method according to claim 4 or 5, wherein the boot image contains a boot.oat file and wherein the step of relocating the boot image according to a new patch delta further comprises:

selecting gaps randomly in a text section of the boot.oat file; and

patching the boot.oat file according to the selected gaps.

7. The boot image randomisation method according to claim 4 or claim 5, wherein the step of loading the cached boot image in the image space in response to the boot image is available in the cache path further comprises:

reading boot.oat from the cached boot image;

adding random gaps in a text section in the boot.oat file and modifying the offsets related to text section accordingly.

8. A boot image randomisation method for an Android RunTime (ART) virtual machine during a system boot comprising:

creating an image space in an ART heap, wherein the image space is used to load a boot image containing a boot.oat file;

checking whether the boot image is cached in a cache path;

relocating the boot image according to a new patch delta in response to the boot image is not available in the cache path by removing all applications' OAT caches, selecting a patch delta randomly, patching and writing the boot image to a path for caching, executing an image space initialization and loading the boot image to the image space; and

wherein the step of relocating the boot image according to a new patch delta further comprises:

selecting gaps randomly in a text section of the boot.oat file; and

patching the boot.oat file according to the selected gaps.

9. A boot image randomisation method for an Android RunTime (ART) virtual machine during a system boot comprising:

creating an image space in an ART heap, wherein the image space is used to load a boot image containing a boot.oat file;

checking whether the boot image is cached in a cache path;

loading the cached boot image in the image space in response to the boot image is available in the cache path;

relocating the boot image according to a new patch delta in response to the boot image is not available in the cache path by removing all applications' OAT caches, selecting a patch delta randomly, patching and writing the boot image to a path for caching, executing an image space initialization and loading the boot image to the image space; and

wherein the step of loading the cached boot image in the image space in response to the boot image is available in the cache path further comprises:

reading boot.oat from the cached boot image;

adding random gaps in a text section in the boot.oat file and modifying the offsets related to text section accordingly.

10. A device comprising:

a processor; non-transitory memory; and instructions stored on the non-transitory memory executable by the processor, for an Android RunTime (ART) virtual machine in a system boot, to:

create an image space in an ART heap; wherein the image space is used to load a boot image;

delete the boot image in a cache path;

relocate the boot image according to a new patch delta in response to the boot image is not available in the cache path by removing all applications' OAT caches, select a patch delta randomly, patch and write the boot image to a path for caching, execute an image space initialization and load the boot image to the image space.

11. The device according to claim 10 further comprises instructions after the instruction to relocate the boot image according to a new patch delta, to:

create a Class Linker according to the ART heap such that when a system class is going to be linked based on the Class linker, an image header will be obtained from the image space in the heap the patched address of boot image will be used to find the system class.

12. The device according to claim 10 or 11, wherein the boot image contains a boot.oat file and wherein the instruction to relocate the boot image according to a new patch delta further comprises instructions to:

select gaps randomly in a text section of the boot.oat file; and

patch the boot.oat file according to the selected gaps.

13. A device comprising:

a processor; non-transitory memory; and instructions stored on the non-transitory memory executable by the processor, for an Android RunTime (ART) virtual machine in a system boot, to:

create an image space in an ART heap, wherein the image space is used to load a boot image;

check whether the boot image is cached in a cache path;

load the cached boot image in the image space in response to the boot image is available in the cache path;

relocate the boot image according to a new patch delta in response to the boot image is not available in the cache path by removing all applications' OAT caches, select a patch delta randomly, patch and write the boot image to a path for caching, execute an image space initialization and load the boot image to the image space;

create a Class Linker according to the ART heap; wherein the instruction to load the cached boot image in the image space in response to the boot image is available in the cache path comprises instructions to:

read an image header from the cached boot image, in which the patch delta for determine loading address is stored;

modify the patch delta randomly to load the boot image in a new address.

14. The device according to claim 13 further comprising instructions to:

amend the Class Linker according to the modified patch delta.

15. The device to claim 13 or claim 14, wherein the boot image contains a boot.oat file and wherein the instruction to relocate the boot image according to a new patch delta further comprises instructions to:

select gaps randomly in a text section of the boot.oat file; and

patch the boot.oat file according to the selected gaps.

16. The device according to claim 13 or claim 14, wherein the instruction to load the cached boot image in the image space in response to the boot image is available in the cache path further comprises instructions to:

read boot.oat from the cached boot image;

add random gaps in a text section in the boot.oat file and modify the offsets related to text section accordingly.

17. A device comprising:

a processor; non-transitory memory; and instructions stored on the non-transitory memory executable by the processor, for an Android RunTime (ART) virtual machine in a system boot, to:

create an image space in an ART heap, wherein the image space is used to load a boot image containing a boot.oat file;

check whether the boot image is cached in a cache path;

relocate the boot image according to a new patch delta in response to the boot image is not available in the cache path by removing all applications' OAT caches, select a patch delta randomly, patch and write the boot image to a path for caching, execute an image space initialization and loading the boot image to the image space; and

wherein the instruction to relocate the boot image according to a new patch delta further comprises instructions to:

select gaps randomly in a text section of the boot.oat file; and

patch the boot.oat file according to the selected gaps.

18. A device comprising:

a processor; non-transitory memory; and instructions stored on the non-transitory memory executable by the processor, for an Android RunTime (ART) virtual machine in a system boot, to:

create an image space in an ART heap, wherein the image space is used to load a boot image containing a boot.oat file;

check whether the boot image is cached in a cache path;

load the cached boot image in the image space in response to the boot image is available in the cache path;

relocate the boot image according to a new patch delta in response to the boot image is not available in the cache path by removing all applications' OAT caches, select a patch delta randomly, patch and write the boot image to a path for caching, execute an image space initialization and load the boot image to the image space; and

wherein the instruction to load the cached boot image in the image space in response to the boot image is available in the cache path further comprises instructions to:

read boot.oat from the cached boot image;

add random gaps in a text section in the boot.oat file and modify the offsets related to text section accordingly.

19. A device running on an Android operating system comprising:

a bootstrap module and a base address randomization module (BARM) wherein the bootstrap module is configured to invoke the BARM upon a system boot, and the BARM is configured to delete a boot image in a cache path and randomise a base address of a boot.oat file in the boot image.

20. The device according to claim 19 wherein the step to randomise the base address of the boot.oat file in the boot image in each system boot comprises the BARM to:

modify a patch delta randomly to load the boot image in a new address.

21. The device according to claim 19 wherein the step to randomise the base address of the boot.oat file in the boot image in each system boot further comprises the BARM to:

modify a Class Linker according to the modified patch delta.

22. The device according to claim 20 further comprising:

a gap randomisation module (GAM) configured to introduce random gaps in a compiled system libraries in the boot.oat file.

23. The device according to claim 22 wherein the step to introduce random gaps in the compiled system libraries in the boot.oat file comprises the GAM to:

monitor and detect when the boot image cache is being created;

in response to detecting the boot image cache is being created,

modify a patchoat module of the Android operating system to randomly pick gaps and patch boot.oat according to the gaps;

modify an offset pointers in the boot.oat file that points to the text section; and

modify the boot.art file to update an OAT end pointer.

24. The device according to claim 20 further comprising a gap randomisation module (GAM) configured to:

monitor and detect when the boot image cache is being created;

in response to detecting the boot image cache is being created,

modify a patchoat module of the Android operating system to randomly pick gaps and patch boot.oat according to the gaps;

modify an offset pointers in the boot.oat file that points to the text section;

modify the boot.art file to update an OAT end pointer and wherein the bootstrap module is further configured to invoke BARM each time after the first system boot and GAM in the first system boot.

25. The device according to claim 20 further comprising a gap randomisation module (GAM) configured to:

monitor and detect for an OAT file being loaded into a memory through a specific function of opening the OAT file during initializing an image space;

in response to detecting the OAT file is being loaded into the memory,

activate a binary rewriting module to reads a boot image from the cache path,

add random gaps in the text section and modify the offsets related to text section accordingly,

load modified boot image into the memory, and wherein the bootstrap module is further configured to invoke BARM and GAM each time after the first system boot.

26. The device according to claim 20 further comprising a gap randomisation module (GAM) configured to:

monitor and detect when the boot image cache is being created;

monitor and detect for an OAT file being loaded into a memory through a specific function of opening the OAT file during initializing an image space;

in response to detecting the boot image cache is being created,

modify a patchoat module of the Android operating system to randomly pick gaps and patch boot.oat according to the gaps;

modify an offset pointers in the boot.oat file that points to the text section; and

modify the boot.art file to update an OAT end pointer

in response to detecting the OAT file is being loaded into the memory,

activate a binary rewriting module to reads a boot image from the cache path,

add random gaps in the text section and modify the offsets related to text section accordingly,

load modified boot image into the memory, and wherein the bootstrap module is further configured to invoke BARM and GAM each time after the first system boot, and

wherein the bootstrap module is further configured to invoke GAM to monitor and detect for an OAT file being loaded into a memory through a specific function of opening the OAT file during initializing an image space in the first system boot, invoke BARM each time after the first system boot and invoke GAM to monitor and detect when the boot image cache is being created.

27. A device running on an Android operating system comprising:

a bootstrap module and a gap randomisation module (GAM) wherein the bootstrap module is configured to invoke the GAM upon a system boot, and the GAM is configured to introduce random gaps in a compiled system libraries in the boot.oat file.

28. The device according to claim 27 wherein the step to introduce random gaps in a compiled system libraries in the boot.oat file comprises the GAM to:

monitor and detect when the boot image cache is being created;

in response to detecting the boot image cache is being created,

modify a patchoat module of the Android operating system to randomly pick gaps and patch boot.oat according to the gaps;

modify an offset pointers in the boot.oat file that points to the text section; and

modify the boot.art file to update an OAT end pointer.

29. The device according to claim 27 wherein the step to introduce random gaps in a compiled system libraries in the boot.oat file comprises the GAM to:

monitor and detect for an OAT file being loaded into a memory through a specific function of opening the OAT file during initializing an image space;

in response to detecting the OAT file is being loaded into the memory,

activate a binary rewriting module to reads a boot image from the cache path,

add random gaps in the text section and modify the offsets related to text section accordingly;

load modified boot image into the memory.