

# Immune Computation of Secure Embedded Linux Core against Viruses and Software Faults

Tao Gong<sup>1,2,3</sup> and Changxing Du<sup>1</sup>

<sup>1</sup> College of Information S. & T., Donghua University, Shanghai 201620, China

<sup>2</sup> Engineering Research Center of Digitized Textile & Fashion Technology, Ministry of Education, Donghua University, Shanghai 201620, China

<sup>3</sup> Department of Computer Science, Purdue University, West Lafayette 47907, USA  
taogong@dhu.edu.cn

## Abstract

*To increase the security of an embedded system, it is important to assure the security of the embedded Linux core in the system. Biological immune system is the core to assure the health of human body and this natural system inspire us to design the security mechanism for the Linux core. To defend the embedded Linux core against viruses and software faults, an immune computation is proposed. First, the embedded Linux is customized from the standard Linux by keeping the Linux core and deleting the unnecessary components. Immunization of the Linux core is designed into the process control, memory management, communication, driving programs, and file system. The artificial immune system of the embedded Linux core is built on the tri-tier immune model, and both viruses and software faults are detected as nonselfs. The selfs are the normal components and the nonselfs are foreign viruses, infected selfs, lost selfs and damaged selfs. This immunization technique will be tested on a prototype of embedded Linux core, by protecting the file system and repairing the damaged files.*

**Keywords:** Immune computation; security; embedded Linux core; viruses; software faults

## 1. Introduction

Unknown viruses are difficult to detect and learn in some security applications [1], and the operation systems such as Windows and Linux have vulnerability to the viruses and attacks. To repair the vulnerability, users need update the operation systems online to set up some new security patches. This updating often provides a new chance for the viruses and attacks to spread the damage through the network. So threat modeling is used to expose some circumstances or events having the potential to cause harm to a system in the form of destruction, disclosure, modification of data, and/or denial of service, and results in a vulnerability assessment [2]. Similar to the threat model, the immune danger theory was proposed by Matzinger [3], and in this danger theory immune response distinguishes the danger signals that are generated by damaged cells. In the embedded Linux core, the threats are the damaged selfs and the foreign non-selfs such as the blackhole attacks [4] and the virus-based wormhole attacks [5], so the threats are the non-selfs in nature. First, the blackhole attacks can transmit malicious broadcast information from a node that the node has the shortest path to the destination aiming to intercept messages. And the wormhole attacks can record packets at one location in the network, tunnel them to other locations, and retransmit them there into the network via viruses.

In fact, the human immune system has another different and advanced security approach to protect the body [6], and this immunization approach emphasize more on selfs (i.e. normal components such as normal cells, immune cells, and antibodies) than the nonselfs such as viruses and cancer cells.

Inspired from this natural powerful security system, a new idea of immunizing the embedded Linux core is proposed to build the normal models of selfs and defend this core against the viruses and software faults, in this paper.

## 2. Embedded Linux Core

Standard Linux is a complex operation system, which has too many components to be set up in embedded systems. So this standard Linux needs trimming to be customized in the embedded systems, and the Linux core should be revised and compiled. The embedded Linux core has some necessary files and directories, as shown in Figure 1.

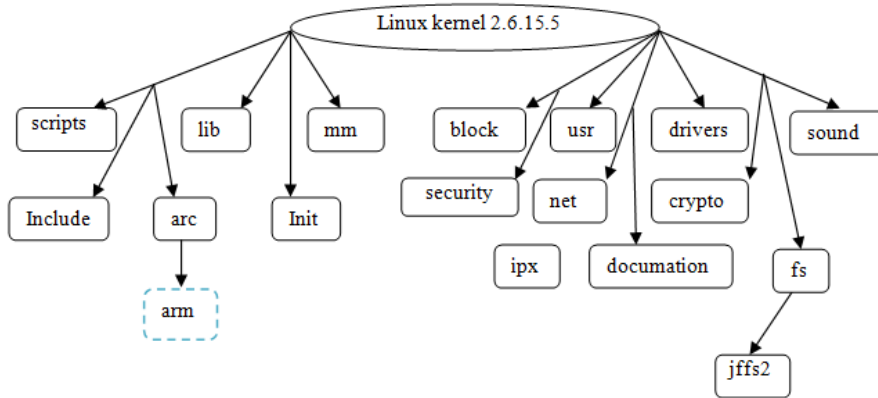


Figure 1. Embedded Linux Core

## 3. Immune Computation of Embedded Linux Core

The immunization of the embedded Linux core is made in process control, memory management, communication, drivers and file system.

### 3.1. Immunizing Process Control

The embedded Linux core utilizes various data structures to organize the system processes in different ways, according to various requirements of the process control sub-system. Each process has its unique identification number (PID), and the immunity of the process control protects the core data structures, which the processes use. At the same time, the creation and elimination of each process are monitored in the core space, in order to avoid the creation of illegal process and prohibit from eliminating the core processes.

### 3.2. Immunizing Memory Management

Memory is one of the most important resources in control of the Linux core, and the memory management sub-system is the most important and difficult part of operation system. The correct memory allocation is a necessary condition for running the normal system. The two modules of the memory management include the mapping from a physical address to the virtual address and the management for allocating the memory of the Linux core via the slab.

The immunity of memory management is used to decline and eliminate the process nonselfs, and this immunity protects the core data structures of memory management to avoid data faults such as data overflow.

### 3.3. Immunizing Root File System

The root file system is a necessary file system for running the Linux operation system. This root file system is stored in the Flash storage as shown in Figure 2, and this storage is divided into multiple partitions such as partition 1, partition 2, and partition 3 etc.

Partition 1 Core mirror	Partition 2 Root file system	Other partition Other file system (such as user space)
-------------------------------	------------------------------------	--------------------------------------------------------------

**Figure 2. Partition Structure of the FLASH Storage in Linux**

### 3.4. Immunizing Special Drivers of Embedded Systems

Driver design is an important step to develop embedded systems, and the drivers provide the interfaces for the applications to control the hardware. The normal embedded systems are based on the normal states of the drivers. The immunity of the driver is used to monitor the normal state of the driver and assure the proper output of the diver. When the driver is abnormal, the immune programs can repair the damaged driver quickly to increase the life of the embedded system.

### 3.5. Immunizing Network Communication

The embedded systems often communicate with other embedded systems or foreign networks, and the security is not only based on the security of the communication protocol, but also relative with the security of the Linux core. If the immune Linux core can detect the data packages with such nonselfs as viruses by detecting the selfs of this core first, this communication will be more secure. This is just the mission of the immunity for network communication.

### 3.6. Immunizing User Data Space

In many embedded systems, the data spaces of users are private, so the immunity of the Linux core protects this privacy. Only the selfs with legal authorization can make access to the private data, and such nonselfs as the viruses with illegal access to the private data are isolated and eliminated by the immune programs. When some data are lost in the storage, these immune programs can repair these data, so the security of this user data space is increased.

On the other hand, the immune programs may increase the costs of the embedded systems in space and time, so the design of immune algorithm is based on the load balance of the embedded system.

## 4. Compiling and Testing Immune Linux Core

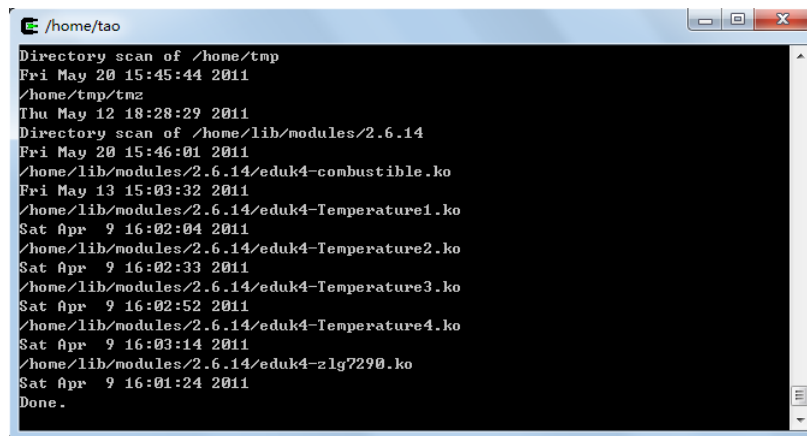
The immune Linux core is compiled according to the regular compiling method, and the step for compiling the immune programs is just after the step for building the normal model of the embedded system. After the immune programs are compiled, the

files Makefile and Konfig are revised and generated. The mirror of the Linux core uses the zImage mirror, and this mirror can be downloaded into the development board to test.

This test includes the following modules:

- (1) Module for repairing the damaged files;
- (2) Module for detecting and eliminating the illegal memory access;
- (3) Module for immunizing the data packages that are received from other embedded systems or networks;
- (4) Module for immunizing the root file system;
- (5) Module for protecting the data spaces for users.

For example, an anti-disaster embedded system has an immune program tmz, and this file is at /home/tmp/tmz, as shown in Figure 3.



```
~/home/tao
Directory scan of /home/tmp
Fri May 20 15:45:44 2011
/home/tmp/tmz
Thu May 12 18:28:29 2011
Directory scan of /home/lib/modules/2.6.14
Fri May 20 15:46:01 2011
/home/lib/modules/2.6.14/eduk4-combustible.ko
Fri May 13 15:03:32 2011
/home/lib/modules/2.6.14/eduk4-Temperature1.ko
Sat Apr 9 16:02:04 2011
/home/lib/modules/2.6.14/eduk4-Temperature2.ko
Sat Apr 9 16:02:33 2011
/home/lib/modules/2.6.14/eduk4-Temperature3.ko
Sat Apr 9 16:02:52 2011
/home/lib/modules/2.6.14/eduk4-Temperature4.ko
Sat Apr 9 16:03:14 2011
/home/lib/modules/2.6.14/eduk4-zlg7290.ko
Sat Apr 9 16:01:24 2011
Done.
```

**Figure 3. Building the Norml Model for the Selves of the Anti-disaster Embedded System**

In Figure 3, the last revision time of the normal immune program file is Thu May 12 18:28:29 2011. Other files are at the lib directory, which stores the driver files.

After the normal model is built, the space-time properties of all the normal components in the embedded system are stored into the self database. Based on the self detection with the normal model, the viruses and the software faults can be detected quickly and accurately, as shown in Figure 4.

For the normal anti-disaster embedded system, all the components of the software are normal, and so they need no repairing. When some files or directories are lost, the software of the embedded system cannot work normally, and so the immune program repairs the lost components with their backup ones, as shown in Figure 5.

Not only the components are perhaps lost, but also may be attacked by such nonselfs as viruses. Then the immune program detects the damaged selves and repairs them with their backup ones, as shown as in Figure 6. For instance, tmz.vbs is a worm program, which is written by VBScript as a kind of virus and used to infect other systems and machines.

```
/home/tao
Directory scan of /home/tmp
Fri May 20 15:45:44 2011
/home/tmp/tmz
Thu May 12 18:28:29 2011
Directory scan of /home/lib/modules/2.6.14
Fri May 20 15:46:01 2011
/home/lib/modules/2.6.14/educ4-combustible.ko
Fri May 13 15:03:32 2011
/home/lib/modules/2.6.14/educ4-Temperature1.ko
Sat Apr 9 16:02:04 2011
/home/lib/modules/2.6.14/educ4-Temperature2.ko
Sat Apr 9 16:02:33 2011
/home/lib/modules/2.6.14/educ4-Temperature3.ko
Sat Apr 9 16:02:52 2011
/home/lib/modules/2.6.14/educ4-Temperature4.ko
Sat Apr 9 16:03:14 2011
/home/lib/modules/2.6.14/educ4-zlg7290.ko
Sat Apr 9 16:01:24 2011
Done.
```

Figure 4. Self Detection in the Normal Embedded System

```
No any nonself has been detected.
/home/tmp
Directory or file /home/tmp exists and need no repairing.
/home/tmp/tmz
Directory or file /home/tmp/tmz doesn't exist and need repairing.
The missing self directory or file is being repaired .....
tmzFrom: /home/backup/tmp/tmz. To: /home/tmp
The missing self directory or file has been repaired successfully!
/home/lib/modules/2.6.14
Directory or file /home/lib/modules/2.6.14 exists and need no repairing.
/home/lib/modules/2.6.14/educ4-combustible.ko
Directory or file /home/lib/modules/2.6.14/educ4-combustible.ko exists and need
no repairing.
/home/lib/modules/2.6.14/educ4-Temperature1.ko
Directory or file /home/lib/modules/2.6.14/educ4-Temperature1.ko exists and need
no repairing.
/home/lib/modules/2.6.14/educ4-Temperature2.ko
Directory or file /home/lib/modules/2.6.14/educ4-Temperature2.ko exists and need
no repairing.
/home/lib/modules/2.6.14/educ4-Temperature3.ko
```

Figure 5. Repairing the Lost in the Embedded System

```
/home/lib/modules/2.6.14/educ4-combustible.ko is a self!
/home/tmp
Fri May 20 15:45:44 2011
/home/tmp/tmz
Thu May 12 18:28:29 2011
/home/lib/modules/2.6.14
Fri May 20 15:46:01 2011
/home/lib/modules/2.6.14/educ4-combustible.ko
Fri May 13 15:03:32 2011
/home/lib/modules/2.6.14/educ4-Temperature1.ko
Sat Apr 9 16:02:04 2011
/home/lib/modules/2.6.14/educ4-Temperature1.ko is a nonself, which was a self.
The nonself file has been eliminated!
The damaged self file is being repaired .....
The damaged self file has been repaired successfully!
```

Figure 6. Anti-virus Immunization in the Embedded System

## 5. Conclusions

The embedded Linux core is very important for many applications of the embedded systems, and the immunization of this embedded Linux core is important for its security. First, the normal model is useful to build the selfs and identify the normal state of this embedded system by the space-time properties. Then, based on this normal model, the viruses and software faults can be detected quickly and accurately, so the immunization can be created for the embedded system of such modules as process control, memory management, communication, drivers and file system etc.

## Acknowledgements

The work was supported in part by grants from the National Natural Science Foundation of China (60874113), Shanghai Educational Development Foundation (2007CG42), and Natural Science Foundation of Shanghai (08ZR1400400). I thank anonymous reviewers and IEEE Fellow Bharat Bhargava and IEEE Senior Member Zixing Cai for their good advice on improving the research in this paper.

## References

- [1] O. Sukwong, H. S. Kim and J. C. Hoe, "Commercial antivirus software effectiveness: an empirical study. IEEE Computer, Vol. 44, No. 3, (2011), pp. 63-70.
- [2] IEEE Std 1074-2006. IEEE Standard for Developing a Software Project Life Cycle Process. 10.1109/IEEESTD.2006.219190 (2006), pp. 1-104.
- [3] P. Matzinger, "The danger model: a renewed sense of self", Science, Vol. 12, (2002), pp. 301-305.
- [4] S. Ramaswamy, H. Fu and K. Nygard, "Effect of cooperative blackhole attack on mobile ad hoc networks", In: ICWN (2005).
- [5] W. Wang, B. Bhargava, Y. Lu and X. Wu, "Defending against wormhole attacks in mobile ad hoc networks", Wireless Communications & Mobile Computing, Vol. 6, No. 4, (2006), pp. 483-503.
- [6] T. Gong and Z. X. Cai, "Artificial immune system based on normal model and its applications", Beijing: Tsinghua University Press, (2011).

## Authors



### Tao Gong

Dr. Gong received the MS degree in Pattern Recognition and Intelligent Systems and PhD degree in Computer Science from the Central South University respectively in 2003 and 2007. He is an associate professor of computer sciences at Donghua University, China, and he was a visiting scholar at Department of Computer Science and CERIAS, Purdue University, USA. He is the General Editors-in-Chief of the first leading journal Immune Computation in its field, and an editorial board member of some international journals such as Journal of Computers in Mathematics and Science Teaching, International Journal of Security and Its Applications, and International Journal of Multimedia and Ubiquitous Engineering. He is a Technical Committee Chair of ISEEIP 2012, and a Publicity Chair of ISA 2012. He is also a program committee member of some international conferences such as IEEE ICNC 2011, IEEE BMEI 2011, WMSE 2011, ICARIS 2012, AITS 2012, CCA 2012, ASP 2012, IST 2012, ISA 2012 and SIS2013 etc. He is a Life Member of Sigma Xi, The Scientific Research Society, a Vice-Chair of IEEE Computer Society Task Force on Artificial Immune Systems, and Chen Guang Scholar of Shanghai. He has published over 70 papers in referred journals and international conferences, and over 20 books such as Artificial Immune System Based on Normal Model and Its Applications etc. He is also a committee member of intelligent robots committee and natural computing committee in the Association of Artificial Intelligence of China.