Security of Distribution Mechanisms for Linux and BSD Operating Systems

Gabriel Ewing
Department of Electrical Engineering and Computer Science
Case Western Reserve University
Cleveland, Ohio

Kevin Nash
Department of Electrical Engineering and Computer Science
Case Western Reserve University
Cleveland, Ohio

***Abstract*—The abstract goes here.**

I. Introduction

An operating system is a piece of software that manages computer hardware resources and provides a variety of services for computer programs. The central core of an operating system, its kernel, is the first layer above hardware itself. Due to of the depth of their functionality, compromised operating systems can potentially yield a great deal more power to an attacker than application software.

Operating systems can be compromised by malicious computer software, such as rootkits, in the course of normal operation. However, attacks can sometimes be launched more easily against the distribution process itself. This can be done in such a way that users unknowingly install a modified version of the expected operating system. If successful, attacks that result in the distribution of a compromised operating system can be both difficult to detect and powerful.

The open-source software model possesses a somewhat different attack surface that its closed- or shared-source counterparts. Open-source operating systems often have much smaller core development teams than popular commercial systems such as Windows and OS X. Open-source operating systems are rarely distributed using physical media, which is the primary distribution mechanism for Windows. The commercial interfaces that are requisite for online distribution of proprietary software also have advantages and disadvantages in security that are different from the security considerations that open-source distributors must account for.

...Add more here...

II. Distributing an Operating System

Body of text goes here...

A. Building a Release

Body of text goes here...

1) Compiling into ISO: Body of text goes here...

2) Overseeing Release: Body of text goes here...

B. Mirroring a Release

Body of text goes here...

1) Mirror Qualifications: Body of text goes here...

2) Fetching Release: Rsync, Zsync...

C. Distribution Methods

Body of text goes here...

1) HTTP: Body of text goes here...

2) FTP: Body of text goes here...

3) BitTorrent: Body of text goes here...

4) Physical Media: Body of text goes here...

III. Attacks on Distribution

Body of text goes here...

A. "Attack One"

Body of text goes here...

1) Notable Usage: Body of text goes here...

2) Countermeasures: N operating systems currently implement these countermeasures, including Foo, Bar, Baz...

Visual aid goes here

B. "Attack Two"

Body of text goes here...

1) Notable Usage: Body of text goes here...

2) Countermeasures: N operating systems currently implement these countermeasures, including Foo, Bar, Baz...

Visual aid goes here

IV. External Risks

Body of text goes here...

A. Re-Hosting and Ownership Hijacking

Body of text goes here...

1) SourceForge: Body of text goes here...

V. Best Consumer Practices

A. Choosing a Protocol

Body of text goes here...

B. Verifying Mirrors

Body of text goes here...

C. Building a Web of Trust

Body of text goes here...

D. "Soft" Risk Mitigation

Sometimes it is best to rely on proven-stable releases. It can be harmful to be on the bleeding edge of development, although it is a service to the industry.

VI. Our Implementations

Body of text goes here...

VII. Conclusion

The conclusion goes here.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.