Protecting Software Intellectual Property Against Counterfeiting and Piracy

Karen Mercedes Goertzel, Booz Allen Hamilton

Abstract. Software counterfeiting and piracy are problems of global proportions that violate the enforcement of Intellectual Property Rights (IPR) of software developers and vendors, and thus threaten their market viability. Moreover, software counterfeiting provides violators with the opportunity to modify and augment the duplicated software code in undesirable ways, including insertion of malicious logic, backdoors, and exploitable vulnerabilities. Technological solutions to these challenges focus on making software authenticity easier to verify, making software more difficult to counterfeit, and making software distribution processes harder to subvert. But the software industry and governments worldwide recognize that technology is not the sole answer to reducing piracy and counterfeiting. They are focusing their efforts both on technological research but even more on IPR legislation, trade agreements, enforcement, "best practices" and awareness that both complement and reinforce technological approaches.

In DoD, DHS, the intelligence community, and other security-focused organizations, a frequently discussed supply chain threat to software is the rogue developer or distributor who embeds malicious logic, backdoors, or intentional vulnerabilities in source code or binary executables prior to releasing them to customers. (From here on "malicious code" should be understood to refer collectively to all three types of inclusions.)

The threat of counterfeiting is another concern, especially counterfeit integrated circuits and network devices, and increasingly software. Three factors drive this concern: (1) counterfeits are often found to be less dependable than the original products they are intended to copy; (2) counterfeiting occurs outside the legitimate supply chain, thus obscuring supply chain transparency and traceability of product, product pedigree, and product provenance; and (3) counterfeiting violates the IPR of the copied product's legitimate vendor/developer and threatens their business viability.

For software, the terms "counterfeiting" and "piracy" are often used interchangeably, so the distinction between them has become obscure. For purposes of this article, piracy means the act of illicitly distributing copies of software without a legitimate license. Counterfeiting means the substitution at any point in the supply chain of legitimate license-bearing software with an illicit replica. Counterfeit replicas are often modified or augmented, e.g., through removal of anti-tamper protections or insertion of malicious code, thus tying the first concern about malicious code with the second about illicit copies. Pirated software is sometimes, but not always, counterfeit. Sometimes it is directly copied without alteration.

Given the three factors cited above, the desire to know a software product's true pedigree (origin) and provenance (tools/processes used to develop and move it through its supply chain) exceed the imperatives of IPR enforcement, also enabling the purchaser to assess the trustworthiness of the parties known to be involved in creating and distributing all components of the software product. From this knowledge can be inferred some level of confidence in the software's dependability, trustworthiness, and authenticity. Pedigree/provenance indicators also provide an "audit trail" by which accountability can be traced back to those parties.

Currently, software pedigree analysis relies on "fingerprints" or "signatures" of acquired source code, which are compared against those of known, legitimate open source or (where applicable) closed source code. A pedigree analysis tool determines whether a given instance of source code has been copied from a known code base and is compliant with the code base's licensing restrictions. The tools also reveal whether the copied code was modified (e.g., through malicious insertions). Palamida, Protecode, and Blackduck Software offer commercial toolsets and/or services for source code pedigree analysis.

Theoretically pedigree analysis should be possible for source code derived through reverse engineering of binary executables. In practice, however, reverse engineering produces reconstituted source code that deviates significantly from the original code-as-written. Any "fingerprint" of the reconstituted source code will be useless as a basis for comparison with the original code. Moreover, use of any pedigree analysis approach that requires reverse engineering would require a vendor willing to: (1) provide access to their source code (as a basis of comparison); and (2) permit the violation of their software license restrictions that prohibit reverse engineering of their executables. Neither of these is likely.

In *Software Piracy Exposed*, the authors state that, "Insiders are constant suppliers to the piracy market" [1]. Digital piracy (warez) groups constantly seek cooperative software industry (as well as music, games, and movie industry) insiders to act as "suppliers" who will copy and upload their employers' software products to warez sites for other group members to download. Moreover, every warez group has at least one industry insider actively working for it. Often, these insiders learn about new releases of products well before those products are officially released, and provide illicit pre-release copies to the warez sites. Pirated pre-release versions of software are often detected by their vendors because they include CD keys or serial numbers that get stripped out of official release distributions.

Another source of pirated software is the low-wage worker in a media packaging warehouse, easily induced by promises of payment to steal and provide CDs to warez groups; a premium is paid for master replication discs.

Software counterfeiting extends to manuals, brochures, labels, certificates of authenticity, and license agreements that come with software products. Direct copies of software are considered counterfeit if any of these ancillary materials is counterfeit. Counterfeiting of ancillary materials may involve copying and slightly altering legitimate product data or substituting persuasive false data that obscures a product's pedigree or hides a negative aspect of the original or counterfeit version. For example, a product that has no vendor or third-party certification (e.g., Open Group certifications, Novell YES certification, etc.) may be distributed by a counterfeiter with one or more forged certificates.

The supplier of a counterfeited/pirated product is under no obligation to support customers who wittingly or unwittingly buy a counterfeited/pirated copy. In some cases, however, legitimate suppliers may be duped into supporting an illicit copy if it includes a valid serial number that is not yet registered. In such cases, the illicit copy's owner can register that serial number and obtain support.

Other concerns raised by counterfeiting/piracy include the potential loss of a legitimate vendor's reputation and brand integrity. The convolution and complexity of the Information and Communications Technologies (ICT) supply chain is such that even original manufacturers may receive counterfeit parts/components or pirated software copies from their suppliers, and may integrate them into their larger products, so the result is a product that is a hybrid of genuine and counterfeit content. If such products are discovered by customers to be defective, or to include malicious insertions, the original manufacturer will be at fault, regardless of its lack of direct responsibility for the counterfeiting or piracy. Manufacturers may need to be even more vigilant as acquirers of ICT parts/components than purchasers of their end products are.

Counterfeits are also frequently offered by independent distributors, brokers, and other "gray markets," which represent "channel diversion" from the legitimate market. This is a particular risk when customers consider lowest price their first or only criterion for source selection. Whenever such customers purchase illegitimate products from gray market sources, the market share of the original manufacturers of legitimate products, and their authorized distributors, is eroded. The financial health of legitimate suppliers is a concern, because product obsolescence (including when manufacturers go out of business) is a major driver for the counterfeiting industry. Indeed, gray market outlets are often the only available sources for obsolete parts.

Gray markets are usually unintentional violators, such as unauthorized distributors/retailers, online auction sites, open source and software reuse repositories, third-party download sites, thrift shops, garage sales, dollar stores, and other low-price retailers. By contrast, black markets are intentional IPR violators. Example black market outlets are software piracy/warez download and peer-to-peer file sharing sites, as well as hackers that use cross-site scripting to surreptitiously redirect customer browsers away from legitimate software download sites to illegitimate, usually malicious, replica sites.

Protecting Against Counterfeiting and Piracy

The software vulnerability most often blamed for making counterfeiting and piracy possible is lack of robust software copy protections. Also decried is a lack of anti-tamper protections to prevent modification or augmentation of copied code. Tamper proofing and tamper deterrence can be expensive to implement, so products mainly aimed at the consumer market are generally limited to having anti-tamper mechanisms applied to their packages, but not to the software itself. By contrast, software intended for the business market is more likely to include direct tamper deterrence and evidence mechanisms.

Whether distributed on physical media or via a digital transmission or download, software, its packaging, its storage facility (warehouse or download server), and its distribution channel (physical transportation mechanisms or network-based download/transmission mechanisms) need to be secured from end to end.

Trusted distribution for software should:

* Enable customers to authenticate the supply source. For digital downloads, the customer should be able to detect whether a cross-site scripting attack has surreptitiously rerouted the browser to a counterfeit download server.
* Tamper proof the software (via hash, digital code signature, and/or digital watermark).
* Include tamper deterrence/evidence mechanisms in physical packaging.
* Use read-only media for shipping software and documentation.
* Authenticate the acquirer, ideally before a software download (e.g., through use of supplier-provided download key), but definitely at installation time and/or first execution time (through entry of supplier-provided installation and execution keys). Increasingly, software executables are encrypted before download or copying to distribution media, so only licensed customers who have the vendor-provided cryptokey can decrypt the software.
* Separate distribution channels for the software and any keys needed to decrypt, install, or execute it. For example, the supplier may require an online buyer to provide an e-mail address to which such keys will be sent. Authenticated distribution channels and separation of paths can be accomplished for physical distributions using registered mail (or a shipping service) and requiring the recipient to sign for the package containing the software. Associated keys should be sent in a separate physical shipment (ideally by a different shipping service).
* Make sure servers and network connections/sessions involved are similarly protected using cryptographic and access control means because online software sales and download transactions are sus ceptible to the same vulnerabilities and threats as all other net worked computer-based online purchasing transactions.

In 1994, the Bellcore Trusted Software Integrity system [2] established the use of a combined cryptography hash function with an X.509 digital signature affixed to a software executable to protect the integrity of that software during its distribution over the Internet. Today, much the same approach is used to protect integrity of software distributions, but the additional concern over software piracy means that digital signatures and hashes are increasingly augmented with software IP protections such as code obfuscation and software watermarking.

Digital asset management systems are used to track, with accountability, the movement of and access to software assets. Such systems employ digital watermarks and other techniques to capture the time and location of every access. Digital software watermarking involves steganographic hiding of steganographic messages (signals) in the software code. Watermarks enable the purchaser to verify the authenticity of the software's supplier and to detect (though not deter) tampering [3]. If a watermark is unique to a particular copy of the code, it also acts as a fingerprint which, if detected on another copy of the software, indicates that one of the two copies was pirated.

Researchers are investigating strong watermarking techniques to degrade performance or prevent execution of illicit software copies. Software watermarking technology thus also provides a means of license enforcement.

In the movie industry, strong watermarks are applied to film content that will be digitally distributed to prevent illicit copying. If a strong watermark is removed or tampered with before the illicit copy is made, the film content will be altered enough to significantly degrade viewing quality. Researchers are investigating strong watermarking techniques to degrade performance or prevent execution of illicit software copies. Software watermarking technology thus also provides a means of license enforcement.

To deter reverse engineering of software as the basis for counterfeiting-related augmentation/modification, code obfuscation is often used. Code obfuscations can be applied to source code, bytecode, object code, or binary executable code; they are alterations that obscure the purpose of the code without changing its operation. The goal is to make it difficult for a would-be reverse engineer to understand or tamper with the code in a meaningful way, or to locate and circumvent any copy-protection mechanisms in the code. A related technology, media obfuscation, physically alters the CD or DVD on which software is distributed, usually by adding a hard-to-reproduce cryptographic "taint" to the disc as it is pressed. This taint must be detected and decrypted for the software to be installable and executable. An example of an obfuscation system for software media is MLS LaserLock.

Digital code signatures applied to software executables do double duty as product authenticators and tamper indicators. Code signatures do nothing to warrant the absence of tampering or malicious code insertions that occur before the code signature is applied.

Digital Rights Management (DRM) controls are increasingly applied to software documentation, especially when delivered in PDF format. DRM embeds or overlays cryptographic access controls on protected digital content, most often to prevent cut-and-paste copying and/or printing by anyone who does not possess the necessary decryption keys. DRM is problematic when used to protect software code [4] because its white box cryptography model renders the digital certificates containing the DRM decryption keys vulnerable after they are issued. Anyone who obtains a copy of the certificate can use it to decrypt the software. This is a particular problem because DRM-protected software is often distributed without any other form of protection such as code signatures. Pirates and counterfeiters often copy the contents of DRM digital signatures and include the extracted decryption keys with the copied software; these keys can be forwarded on with each copy made. DRM products also require the protected software's installer or user to enter and verify long alphanumeric installation keys or passphrases, making it impossible to fully automate software installation.

Tethering (a.k.a. "product activation") is used by some major software vendors (e.g., Microsoft, Adobe) to enforce software licensing and prevent piracy. Tethering links the software's license with the specific computer hardware on which it is first installed. The software records the unique identifier of the computer's CPU or Ethernet card's MAC address. The hardware identifier is then hashed and sent to the vendor's product activation server, which uses a non-invertible hash function to generate a second unique identifier that it returns to the software. The software can only execute if both hash identifiers are present. Copying the installed software to another CPU will necessarily "break" the hardware-software binding and cause execution to fail. Tethering evolved from the "dongle" concept. A dongle was a device required by some software vendors in the 1990s and early 2000s; it had to be connected to a bus on the computer before the installed software could execute. If the dongle could not be detected by the software's boot routine, execution would fail. Dongles were abandoned by most software vendors due to inconvenience and unreliability.

Some vendors now ship software with a hardware or software guard module that monitors the running software to ensure its authenticity and integrity. Software guards can be embedded in the software code itself, and can perform simple tasks such as checksum calculation/validation and code repair. Recently, vendors have been implementing groups of cooperating guards that work in tandem to perform more sophisticated security tasks. For example, a software product may have different parts of its code protected by guards that apply different checksums to different parts of the software. If an attacker manages to crack one checksum, the others will remain intact unless they are all cracked as well. The time and effort needed to crack so many checksums may be enough to act as a deterrent to most hackers.

Some software products include monitoring programs that perform execution-time checks to ensure the software has been legitimately licensed. The monitor reports any license violations to the software vendor or a third party. Such monitoring programs often perform additional spyware functions, such as tracking and reporting user activities to advertisers who have paid the software vendor for use of the vendor's customers' data.

Note that there are inherent weaknesses in cryptographic solutions such as digital signatures and certificates for code signing. Non-white box encryption of code and many other protections depend on the presence of an active network channel between the system on which the software is installed and a server operated by the software vendor (e.g., for certificate validation, host attribute checking, etc.). Such a network dependency poses non-trivial difficulties for software on platforms that operate offline. Add to this the inherent complexity of cryptokey distribution and key and certificate revocation. And finally, all software protection mechanisms cannot be fully trusted because, as with all ICT, there is no way to verify that the tools that implemented/applied the software protection mechanisms were authentic and trustworthy.

IPR Enforcement Efforts

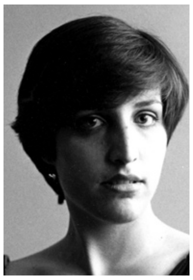
In 2005, KPMG and the Alliance for Gray Market and Counterfeit Abatement published a study that found 10% of information technology products likely to be counterfeits, representing a conservatively estimated loss in revenues and outlays due to returns and exchanges of $10 billion annually [5]. Given the size of potential losses involved, governments and industry trade associations are actively pursuing technology, legislative, regulatory, and trade-related mechanisms for protecting IPR, catching and prosecuting IPR violators, and reducing incidence of counterfeiting and piracy overall. A small but significant portion of this activity focuses on software.

Software counterfeiters and pirates are being caught and prosecuted under increasingly strict IP protection laws thanks to national and international law enforcement efforts such as the multinational Operation Fastlink and U.S. Operation Higher Education. The latter resulted in the confiscation of hundreds of illegal software online distribution hubs; the removal of more than $50 million worth of illegally copied software, games, movies, and music from illicit distribution channels; and the conviction of more than 60 perpetrators in the U.S. alone.

While many prosecuted counterfeiters and pirates are small-scale operations, some are impressively large and organized. For example, from 2005 to 2007 a joint FBI/China Ministry of Public Security law enforcement operation named "Summer Solstice" unearthed an organized crime ring in the U.S. and China responsible for hundreds of thousands of counterfeits of Microsoft, Symantec, and other vendors' software. Summer Solstice operations seized more than 337,000 counterfeit software CDs and certificates of authenticity (estimated retail value $502 million) plus eight high-quality master replication discs and dismantled retail facilities and factories of more than 14 major counterfeiting organizations. Also heartening is the fact that under global pressure to increase enforcement of international IPR laws and agreements, China has finally become more diligent in pursuing software counterfeiters. In 2009, for example, a district court in Suzhou heavily fined and imprisoned four men found guilty of distributing counterfeit versions of Windows XP and other software programs over the Internet [6]. Similar crackdowns have taken place in other countries where counterfeiting and piracy are notorious (e.g., Thailand) [7].

A number of organizations devoted to IPR issues and initiatives have emerged in recent years, including the Federation Against Software Theft and Japan's Association for Copyright of Computer Software. Virtually every trade association representing the software industry (e.g., Business Software Alliance, Software and Information Industry Association, Entertainment Software Association, Content Delivery and Storage Association, Entertainment and Leisure Software Publishers Association) has instituted IPR enforcement, best practice, certification, and awareness efforts. Moreover, broader IPR initiatives, ranging from legislative initiatives (e.g., Piracy and Counterfeiting Amendments Act of 1982, Digital Millennium Copyright Act, Trade Enforcement Act of 2009, Pro-IP Act of 2007, Section 107 of the UK Copyright Designs and Patents Act of 1988) to government awareness and enforcement initiatives (e.g., 1998 Presidential Executive Order 13103 on "Computer Software Piracy," Organization for Economic Cooperation and Development Project on Counterfeiting and Piracy, National IPR Coordination Center, National Intellectual Property Law Enforcement Coordination Council, FBI anti-piracy warnings, NIST National Software Reference Library) to international trade initiatives (e.g., World Trade Organization Agreement on Trade-Related Aspects of IPR, Anti-Counterfeiting Trade Agreement, World Intellectual Property Organization Copyright Treaty of 1996, Asia-Pacific Economic Cooperation Anti-Counterfeiting and Piracy Initiative) all emphasize ICT-related IPR enforcement (hardware and software) as key to continuing the health not just of national economies, but of the global economy.

ABOUT THE AUTHOR



Karen Mercedes Goertzel, CISSP, leads Booz Allen Hamilton's Security Research Service. An expert in the insider threat to information systems, ICT supply chain risk management, software assurance, and cross-domain information sharing, she has performed in-depth research and analysis for customers in DoD, intelligence community, civil agencies, and industry in the U.S., the UK, NATO, Australia, and Canada. She was lead author/editor of the DoD Information Assurance Technology Analysis Center's 2010 State-of-the-Art-Report entitled *Security Risk Management for the Off the Shelf Information and Communications Technology Supply Chain*.

Karen Mercedes Goertzel
Booz Allen Hamilton
c/o P.O. Box 694
Merrifield, VA 22116-0694
Phone: 703-698-7454
E-mail: goertzel_karen@bah.com

REFERENCES

1. Craig, Paul and Ron Honick. Software Piracy Exposed. Burlington, MA: Syngress Publishing, 2005.
2. Rubin, Aviel D. "Trusted Distribution of Software over the Internet", in *Proceedings of the 1995 Internet Society Symposium on Network and Distributed System Security* (San Diego, CA: February 1995), pages 47-53. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.45.4731&rep=rep1&type=ps> (The Betsi technology was also submitted by Rubin as Internet Engineering Task Force Request For Comment 1805, "Location-Independent Data/Software Integrity Protocol", June 1995. <http://tools.ietf.org/html/rfc1805>.)
3. Myles, Ginger. "Using Software Watermarking to Discourage Piracy". *ACM Crossroads*, 10:3 (Spring 2004) <http://www.acm.org/crossroads/xrds10-3/watermarking.html>.
4. Djekic, Petar and Claudia Loebbecke. "Software Piracy Prevention through Digital Rights Management Systems". *Proceedings of the Seventh IEEE International Conference on E-Commerce Technology.* Munich, Germany, July 2005.
5. KPMG and the Alliance for Gray Market and Counterfeit Abatement. "Managing the Risks of Counterfeiting in the Information Technology Industry" (2005) <http://www.agmaglobal.org/press_events/press_docs/Counterfeit_WhitePaper_Final.pdf>.
6. Lau, Justine. "Software Groups Hail China Piracy Verdict". *The Financial Times* (21 August 2009)
7. Einhorn, Bruce. "Microsoft Has Hope in Asian Piracy Fight". *Business Week* (27 February 2009) <http://www.businessweek.com/globalbiz/content/feb2009/gb20090227_551561.htm>.

ADDITIONAL READING

* Atallah, Mikhail J., Eric D. Bryant, and Martin R. Stytz. "A Survey of Anti-Tamper Technologies". *CrossTalk: The Journal of Defense Software Engineering* (November 2004) <http://www.crosstalkonline.org/storage/issue-archives/2004/200411/200411-Atallah.pdf>.
* Castro, Daniel, Richard Bennett, and Scott Andes. "Steal These Policies: Strategies for Reducing Digital Piracy". (December 2009) <http://www.itif.org/index.php?id=324>.
* Collberg, Christian and Jasvir Nagra. *Surreptitious Software: Obfuscation, Watermarking, and Tamperproofing for Software Protection*. Hoboken, NJ: Wiley, 2009.
* Goertzel, Karen Mercedes, Theodore Winograd, et al., for Defense Technical Information Center Information Assurance Technology Analysis Center. *Security Risk Management for the Off-the-Shelf Information and Communications Technology Supply Chain* (Unclassified//Distribution limited to U.S. government and government contractors). Herndon, VA: 2010.
* National Computer Security Center. *Guide to Understanding Trusted Distribution in Trusted Systems* (NCSC-TG-008, December 1988) <http://www.fas.org/irp/nsa/rainbow/tg008.htm>.
* Software Piracy and Measures for Prevention of Piracy. <http://legalsutra.org/665/software-piracy-and-measures-for-prevention>.
* Yacobi, Yacov, Microsoft Research, and Gideon Yaniv, COM Academic Studies. "Counterfeiting and anti-counterfeiting of software and content". *Proceedings of the 8th ACM Workshop on Digital Rights Management*, Alexandria, VA (October 2008): 53–58.