Software on the Witness Stand: What Should It Take for Us to Trust It?

Sergey Bratus[1], Ashlyn Lembree[2], Anna Shubina[1]

[1] Institute for Security, Technology, and Society, Dartmouth College, Hanover, NH
[2] Franklin Pierce Law Center, Concord, NH

1 Motivation

We discuss the growing trend of electronic evidence, created automatically by autonomously running software, being used in both civil and criminal court cases. We discuss *trustworthiness requirements* that we believe should be applied to such software and platforms it runs on. We show that courts tend to regard computer-generated materials as *inherently trustworthy* evidence, ignoring many software and platform trustworthiness problems well known to computer security researchers. We outline the technical challenges in making evidence-generating software trustworthy and the role Trusted Computing can play in addressing them.

This paper is structured as follows: Part I is a case study of electronic evidence in a "file sharing" copyright infringement case, potential trustworthiness issues involved, and ways we believe they should be addressed with state-of-the-art computing practices. Part II is a legal analysis of issues and practices surrounding the use of software-generated evidence by courts.

Part I: The Case Study and Technical Challenges

2 Introduction

Recently the first author was asked to serve as an expert witness in a civil lawsuit, in which the plaintiffs alleged violation of their copyrights by the defendant by way of a peer-to-peer network. Mavis Roy, of Hudson, New Hampshire, had been charged by four record labels with downloading and distributing hundreds of songs from the Internet.

The principal kind of evidence that the plaintiffs provided to the defendant's counsel (the second author), and which, judging by their expert witness' report, they planned to use in court to prove their version of events that implied the defendant's guilt, was a long **print-out of a computer program**.

Furthermore, the timing pattern of the computer program's recorded actions led us to believe that the program produced the print-outs in an **automatic**

fashion rather than as a result of a human operating it *interactively* via a human-computer interface with the operator selecting appropriate actions, stopping to inspect the results, making determinations, forming hypotheses, and planning further actions.[3]

Thus it appears that the *only* entity to "witness" the alleged violations and to produce an account of them for the court – in the form of a series of print-outs – was in fact an autonomous piece of software, programmed by a company acting on behalf of the plaintiffs and RIAA, and running on a computer controlled by this company.

A Sci-Fi writer might say that the program in question was acting as an autonomous "robotic investigator" (or a "robotic witness"), selecting targets for its investigation and recording its investigative actions in the print-outs as evidence to be used in court. We understand that such evidence has already made appearance in many so-called P2P file sharing cases filed by the Recording Industry Association of America (RIAA) across the US.[4]

Clearly, **software entrusted with such an important function must be held to special, higher standards of trustworthiness.** As any computer scientist (and, indeed, any programmer) knows, bugs and misconfigurations are inherent in software, including – despite the programmers' vigorous efforts to the contrary – in mission-critical software, and can be deadly.[5] Defining such standards in a way consistent with the state-of-the-art knowledge of the technical, legal, and social aspects of the problem poses a multi-disciplinary research challenge. In particular, the following aspects — at least — must be considered:

- **Software trustworthiness.** How much can the software be relied on to be error-free and to operate as expected? Such questions are central to Computer Science in general, and to Computer Security in particular, and an acceptable answer should involve a consensus by computer security experts.
- **Trier-of-fact perceptions.** There is a certain common expectation of precision and impartiality associated with computer systems by non-specialists. However, computer practitioners themselves joke that "computers make very fast, very accurate mistakes", and exchange cautionary stories of ubiquitous computer "bugs".[6] This phenomenon of human trust and the potential trier-of-fact bias should be investigated by legal scholars and sociologists.
- **Software as a witness?** Witnesses in court make their statements under oath, with severe consequences of deviating from the truth in their testimony.

[3] As a forensic examiner would do when analyzing a hard drive's contents with software like *Encase* or a network packet trace with software like *Wireshark* that makes no judgments or determinations of its own but merely presents information to the human expert.

[4] For information and defense attorney perspective on these cases see, e.g., `http://recordingindustryvspeople.blogspot.com/` .

[5] E.g., the RISKS Digest `http://catless.ncl.ac.uk/risks` abounds with dramatic examples.

[6] The above-mentioned RISKS Digest is recommended reading in the Computer Security course at Dartmouth College and other leading higher education institutions.

Witnesses are then cross-examined in order to expose any biases or conflicts of interest they might have. Computer-generated evidence comes from an entity that cannot take an oath ensuring its intent of providing the truth (only programmers directly responsible for creating that entity can do so), nor receive an adversarial examination (which would reasonably apply only to the code and function of the software). Ensuring equal responsibilities for "direct" human witnesses and those who are responsible for the creation of the computer-generated evidence requires research by legal scholars.

In this case study we consider the technical aspects of what we believe it should take computer science experts to deem the output of autonomously operating software *trustworthy*, considering both the extreme malleability of such outputs and the need for mitigating the effects of unintended bugs and misconfigurations.

The structure of this case study is as follows. We subdivide and consider the questions posed above, using our experience with the plaintiff's computer-generated evidence in the above-mentioned "file sharing" case, expert opinions of computer scientists in similar previous cases, and other court decisions. Then we explain the connection of the desired trustworthiness properties with the concepts of Trusted Computing (TC) and sketch a design of how TC techniques can help achieve the stated trustworthiness goals.

3 Summary of the Roy Case

Mavis Roy, of Hudson, New Hampshire, had been charged by four record labels with downloading and distributing hundreds of songs from the Internet. The four members of the Recording Industry Association of America (RIAA) brought a case against Roy in U.S. District Court, following a letter from the record companies' attorneys that directed her to a web site where she could pay by credit card to settle the case. Since she did not have a computer in her house at the time she was alleged to have downloaded the music, she ignored the request, thinking it was a scam.[7]

3.1 Case Materials

The subpoena. Ms. Roy's ISP received a subpoena issued by the plaintiff's lawyers. The ISP was asked to identify the subscriber based on an *IP address* and a moment in time (*date, hour, minute, and second*). The ISP disclosed the subscriber account information, including name, phone number, and mailing address.

[7] See the press release of the Franklin Pierce Law Center's Consumer and Commercial Law and Intellectual Property and Transaction Clinics, `http://www.piercelaw.edu/news/posts/2009-06-18-victory-in-downloading-case.php`.

Basis for the subpoena and lawsuit. The materials received by Ms. Roy defense lawyer included, besides the subpoena, printouts of several kinds of software-generated logs and the plaintiff's expert witness report[8] that contained an interpretation of these logs.

The latter report contained statements that the computer with the IP address in question was "registered" to Ms. Roy and engaged in file sharing. The basis for this expression was unclear, since the defendant, as far as we know, never had to register her computer or any specific computer with her ISP (which typically only requires the customer to register the MAC address of the *cable modem* at service activation time), and no MAC addresses at all were present in either the ISP response to the subpoena or any other case documents. Many other statements and conclusions of this and similar plaintiff's expert witness reports have been disputed by expert witnesses[9], but their analysis is beyond the scope of this paper, in which we focus on the content and presentation of the evidence itself.

All logs (titled "Evidence for <number>" were provided as text files in PDF format, summarized in Table 1. The filenames followed the common pattern of `RoyMNH0xxx.PDF`; the first column of the table contains the unique (xxx) part of the filename. Samples from these files are shown in Figures 1– 9.

The choice of format, especially for representing packets as ASCII printouts of their (printable) bytes, complicated analysis of data and introduced additional ambiguity. For example, one can only guess what actual bytes corresponded to non-printable characters, rendered in printouts as a thick black dot; checksumming or cryptographic hashing of such packet captures is impossible, and, as far as we know, was not performed. The most voluminous log (785 pages long, and over 83% of the total submitted pages) contained no relevant information about packets other than their length and was thus of little help for cross-validating other logs.

In particular, the IP addresses contained in such packets – a crucial part of the subpoena – could not be readily verified, nor could other relevant TCP/IP information, such as the Time-To-Live (TTL) values, which could have helped to validate the network path, be readily extracted.

---

[8] *Declaration and Expert Report* by Dr. Doug Jacobson from January 29, 2009.

[9] A selection of such arguments can be found in

1. *Expert witness report* by Dr. J.A. Pouwelse in UMG Recording Inc., et al. v.Lindor, available from `http://www.ilrweb.com/viewILRPDF.asp?filename=umg_lindor_080215ExpertWitnessReportPouwelse`
2. *Declaration* of Jason E. Street in Arista Records, LLC, et al. v.Does 1–11, available from `http://www.ilrweb.com/viewILRPDF.asp?filename=arista_does1-11_070806DeclarationJaysonStreet`
3. *Expert witness report* by Dr. Yongdae Kim in Capitol v. Thomas, available from `http://beckermanlegal.com/pdf/?file=/Lawyer_Copyright_Internet_Law/virgin_thomas_090303DeftsExpertWitnessReport.pdf`

In all of these documents, the assumption that decoding of such information by the generating software was performed without error was apparent. Yet, at least in the case of the document that apparently purported to contain the traced route to the IP in the subpoena, the software obviously failed to operate correctly, as can be see in Figure 7. The reason for this could have been either internal code faults or network configuration faults, or both; we discuss this further in Section 5.2.

xxx:	Purport	Description	Page count
054	"Download Info For <filename>"	ASCII printout of IP packets with IP addresses decoded	124
178	"IP byte log for user at address <IP> for <filename>"	One line per packet: "timestamp, StartByte, %d, EndByte, %d, TotalBytes %d"	785
963	"Shared file matches for user at address <IP:port>"	Filename, length, checksum	1
964	"RECEIVED PACKET <timestamp>"	ASCII printout of IP packet	9
973	"Initializing analysis of user <IP:port>"	Log of actions such as "Attempting to match files", "Choosing files to download", "Initiating download of <filename>"	4
977	"Tracing route to <IP>", "DNS Lookup for <IP>"	Failed traceroute	1
978	"Log for User at address <IP> generated on <timestamp>"	File name and SHA1	11
989	"Total Recognized Files Being Distributed"	File name and size	8

Table 1. Evidence materials in Roy case

3.2 Case Outcome

The case was settled in June 2009. Under the terms of settlement, the case is dismissed with prejudice and neither side is paying the other any money.

4 Witness Trustworthiness: Human vs. "Machine"

Humans' testimony not assumed to be impartial. When human witnesses take the stand, the triers-of-fact are expected to generally consider the possibility that they, despite the oath, may render an untruthful or factually incorrect account of events and circumstances due to a conflict of interest or bias. A possibility

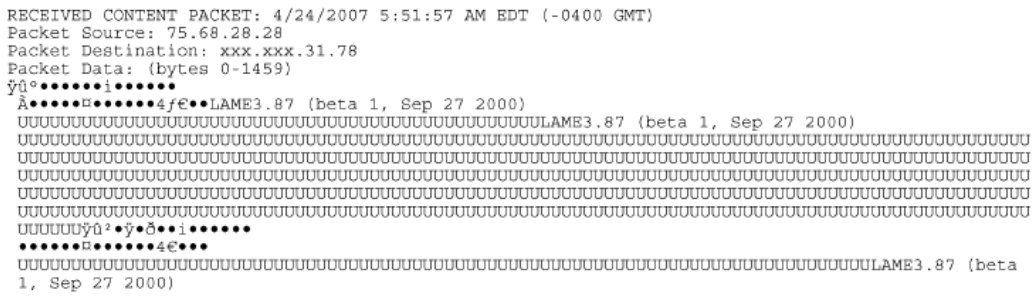

Fig. 1. Sample of RoyMNH054

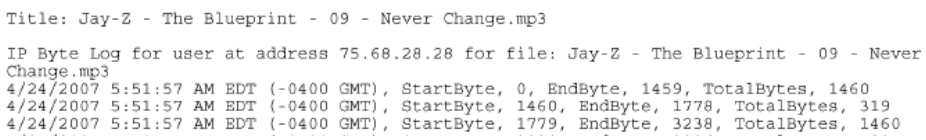

Fig. 2. Sample of RoyMNH0178

of bias may also exist despite the witnesses' genuine desire to render a truthful testimony. Similar considerations apply to expert witnesses.

In short, a human witness' testimony is not automatically assumed to be trustworthy. Specific court procedures such as cross-examination and deposition by the opposing lawyers have evolved for challenging such testimony, in particular, for exposing any potential conflicts of interest to triers-of-fact who, based on their experiences, may or may not consider them significant enough to distrust the testimony.

Biases and conflicts of interest become particularly important in the case of expert witnesses, where triers-of-fact do not have direct and specific knowledge of the subject matter and must therefore rely on the impartiality of the expert's representations, or, in the very least, weigh the relative credibilities of the opposing expert witnesses, as exposed by cross-examination.

The illusion of "machine" impartiality. However, when computer-generated data is introduced as evidence in court, there appears to be a strong assumption that such evidence is somehow impartial and as such more trustworthy than testimony given by a human witness or an expert witness.

For example, in the *UMG vs. Lindor* case, similar to the Roy case we studied, the court seems to have assumed that the discovery request for software, source code, and algorithm were sought solely to address selection of Lindor as a defendant and *not whether the software-generated output could be distrusted or doubted as a complete and objective testimony of events "observed" or caused by the software.*

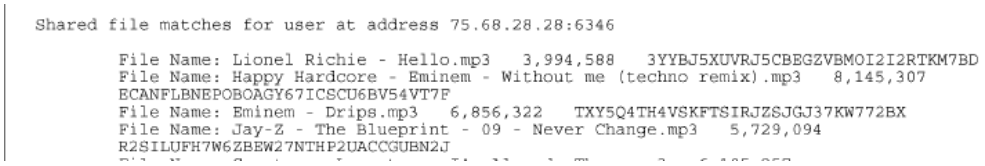

Fig. 3. Sample of RoyMNH0963

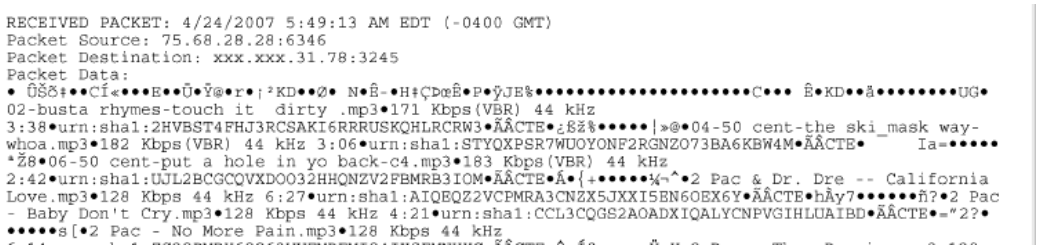

Fig. 4. Sample of RoyMNH0964

In particular, the court concluded that "the software, source code, or algorithm that MediaSentry uses to obtain screen shots is irrelevant to the question of whether the screen shots accurately depict copyright violations that allegedly took place on defendant's internet account."[10]. The court referred to MediaSentry materials as "objective data" and seems to have assumed that "the screen shots attached to the complaint fairly and accurately represent what was on a computer that allegedly was using defendant's internet account at the time of the interception"[11] – essentially because the evidence was generated and presented in a computer format!

It may be inferred that the court *assumed that the software and algorithm were infallible* and therefore fairly and accurately represented what was on [Lindor's] computer.

We can only hypothesize that this comes from the perceived properties of the nature of a "machine" as something that repeatedly and reliably performs mechanical actions, or "computer" as an "idiot savant", in the public mind, as well as from daily experience with commodity electronic devices. As a society, we have been persuaded to trust machines and to rely on them, and thus to view them – despite occasional breakdowns and errors, even dramatic ones – as inherently trustworthy, all things considered.

This attitude ignores the crucial fact that computer software and systems can be and have been programmed and configured to incorporate

[10] UMG Recordings, Inc. v. Lindor, U.S. Dist. Court, E.D.N.Y., Docket No. 05-CV-1095 (May 16, 2008 Order on defendant's motion to compel) ("Lindor Order")

[11] Lindor Order at 5. See also Part II for discussion of Lindor.

```
4/24/2007 5:49:32 AM EDT (-0400 GMT)    Initializing analysis of user 75.68.28.28:6346
(ArchiveID: 760387)
4/24/2007 5:49:32 AM EDT (-0400 GMT)    Rule Name: Rec 2 Gnutella c
4/24/2007 5:49:32 AM EDT (-0400 GMT)    System Build Version: 1.30.3560
4/24/2007 5:49:32 AM EDT (-0400 GMT)    Scanner Name: DC014 (agent ID 323)
4/24/2007 5:49:32 AM EDT (-0400 GMT)    Total Recognized Audio: 218
4/24/2007 5:49:32 AM EDT (-0400 GMT)    Total Recognized Video: 19
4/24/2007 5:49:32 AM EDT (-0400 GMT)    Total Recognized Software: 1
4/24/2007 5:49:32 AM EDT (-0400 GMT)    Total Recognized Documents: 1
4/24/2007 5:49:32 AM EDT (-0400 GMT)    Total Recognized Files Being Distributed: 480
4/24/2007 5:49:32 AM EDT (-0400 GMT)    ================================================

4/24/2007 5:49:44 AM EDT (-0400 GMT)    Connection Type: Direct

4/24/2007 5:49:44 AM EDT (-0400 GMT)    Attempting to match files
4/24/2007 5:50:04 AM EDT (-0400 GMT)    Found Match: Lionel Richie - Hello.mp3
4/24/2007 5:50:11 AM EDT (-0400 GMT)    Found Match: Happy Hardcore - Eminem - Without me
(techno remix).mp3
4/24/2007 5:50:12 AM EDT (-0400 GMT)    Found Match: Eminem - Drips.mp3
```

Fig. 5. Sample of RoyMNH0973

```
4/24/2007 6:14:52 AM EDT (-0400 GMT)    Successful download of Jay-Z - Vol.1 In My Lifetime - 11
- Real Niggaz.mp3
                                         First Packet Received: 4/24/2007 5:54:27 AM EDT (-0400
                                         GMT)
                                         First Download Packet Received: 4/24/2007 5:54:27 AM
                                         EDT (-0400 GMT)
                                         Last Download Packet Received: 4/24/2007 5:56:28 AM EDT
                                         (-0400 GMT)
                                         Last Packet Received: 4/24/2007 5:56:22 AM EDT (-0400
                                         GMT)
                                         Bytes Completed: 4,948,606
                                         Copying file: Jay-Z - Vol.1 In My Lifetime - 11 - Real
                                         Niggaz.mp3
                                         Logging Jay-Z - Vol.1 In My Lifetime - 11 - Real
                                         Niggaz.mp3
```

Fig. 6. Sample of RoyMNH0973-1

biases and malfeasant logic that skewed their functionality and reporting output to suit the interests of their programmer or vendor. In other words, putting a bias or an expression of an ulterior motive into the form of a computer program is not unthinkable; it is not even very hard (but, as we will show, much harder to detect than to commit).

A computer scientist understands that the language of a computer program does not somehow make it impossible for the speaker to "tell a lie", intentionally or unintentionally, but, on the contrary, is as open to malfeasance or honest error (such as programmers' overconfidence) as any other kind of human expression. However, the public perception appears to be that computer technology inherently adds trustworthiness to human activities, by making it harder for the humans involved to distort reality and fall to deception or self-deception.

However, there are dramatic examples to the contrary. For example, according to news reports, the programmer of red light traffic cameras in Italy conspired "with 63 municipal police, 39 local government officials, and the managers of seven different companies in order to rig the system so that it would turn from

```
Evidence for Log Ref ID: 126582810

Tracing route to 75.68.28.28...

1               20ms
2               20ms
3               20ms
4               20ms
5               20ms
6               20ms
7               20ms
8               20ms
9               20ms
10              20ms
11              20ms
12              20ms
13              20ms
14              20ms
15              20ms
16              20ms
17              20ms
18              20ms
19              20ms
20              20ms
21              20ms
22              20ms
23              20ms
24              20ms
25              20ms
26              20ms
27              20ms
28              20ms
29              20ms
30              20ms
Trace complete.

DNS Lookup for 75.68.28.28:

1       c-75-68-28-28.hsd1.nh.comcast.net
DNS lookup complete.
```

Fig. 7. Excerpt from RoyMNH977: failed trace route

yellow to red quicker, therefore catching more motorists."[12]. The intentional, strong bias programmed into the system was only discovered because the unusually high number of reported fines drew an official's suspicion; had the bias been less pronounced, it might have not been detected at all.

Moreover, a bias or deviation from trustworthy behavior need not be malicious or intentional. Programmers and operators may genuinely believe that their systems are operating correctly and as intended, whereas in reality they may be subject to subtle or catastrophic errors. We discuss examples of such errors in the next section.

4.1 The Need for Code Examination

We take the position that **the code of the software must be made available for detailed examination by experts**, especially in such cases as Roy, where *reliability of software-generated evidence cannot be checked or increased by using alternative*

[12] `http://arstechnica.com/tech-policy/news/2009/02/italian-red-light-cameras-rigged-with-shorter-yellow-lights.ars`

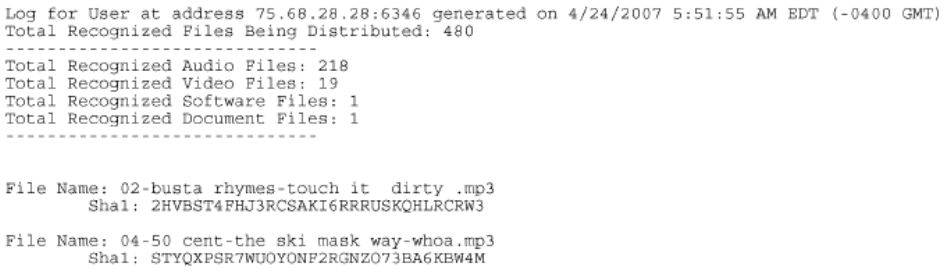

Fig. 8. Sample of RoyMNH0978

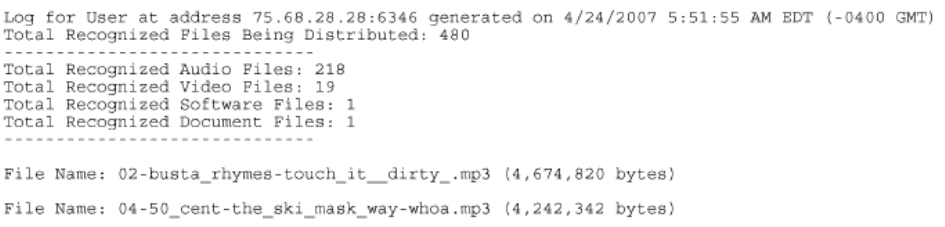

Fig. 9. Sample of RoyMNH0989

resources (e.g., by using competing products for re-testing the same forensic sample, see discussion of the reliability of repeatable vs unrepeatable tests by courts in Part II.).

One important consideration in such cases that involve *transient events* captured only by a single instance of software (and all the more so when the software is plaintiff's) is that the defendants are foreclosed from exonerating themselves by providing independent sources of evidence or causing independent tests to be performed (such as with exonerating DNA evidence). Thus we believe that special-purpose "witness" software that produces a record of transient events must itself be captured in an attestable form tied to the produced output, and its source code examined as explained below.

In cases where a possible conflict of interest is involved (e.g., when the software vendor might profit from a false positive bias or "overdetection" of violations) the code examination must be conducted in great detail to exclude the possibility of subtle bugs resulting in such bias. Although not easy or cheap, such analysis is effective and can be effectively taught (cf. "Hack-the-vote" project [1], in which students at Rice University competed in introducing and detecting such biases into e-voting software).

The Daubert criteria connection to trustworthiness examination of code. Federal courts apply the "Daubert standards" (discussed in Part II) to admissibility

of expert testimony. Considering that autonomous evidence-producing software includes and represents an expression of expert domain knowledge, an analogy can be drawn between the goals of these standards and of code examination for trustworthiness.

Namely, this trustworthiness examination of software should establish:

- *absence of bias*, as discussed above;
- *competence* of the programmers – which can only be conclusively judged via a source code review, the long "invisibility" and subsequent impact of flaws left in legacy binaries by less that competent programmers being notorious;
- *methodology's* reliability – even if competently programmed, the program's algorithms themselves may be flawed.

We briefly discuss the second and third items in the following section, to show that even in the absence of any malicious intent or negligence the trustworthiness of software is not assured. We note that courts have ordered code review in several cases, e.g., State vs. Chun, as discussed in Part II.

4.2 Reasons to Distrust Computer Programs

Many researchers have struggled to come up with techniques for answering the following two questions.

1. Can a computer program be trusted to behave in the desired way?
2. Did a certain computer program produce a certain output?

There are several reasons these problems are hard.

First, programs frequently contain bugs that are hard to find through code inspection (and that may not be detectable without code inspection). An array of techniques and tools have been designed to automatically inspect source code of programs. These techniques and tools range from checkers that detect simple known problems (such as the UNIX utility lint, which, among other things, checks C code for "=" used instead of "==") to *model checkers*, such as SPIN [2], designed for the purpose of detecting concurrency problems when multiple processes interact.

Although these checkers are very useful in detecting certain sets of problems, manual code inspection remains the only way that can, in theory, check for all possible failures. In practice, however, such an exhaustive inspection typically has prohibitive time costs and is likely to overlook the more complicated problems. Many bugs in open source software have existed for months or years, despite examination by the open source community. Complicated attacks on complicated algorithms are generally unpredictable (since, if they were predictable, they would not have been ignored when the algorithms were designed).

Second, the programmers may have made implicit (and incorrect) assumptions about the environment in which the program would be run. Cases where such assumptions led to real-world failures are described in nearly every issue of RISKS digest.

Third, the program may have either been malicious from the start or subverted by an attacker. If the program was modified or replaced, code examination would be of little use in deciding on its trustworthiness.

Finally, code inspection may be of no use if the compiler, the interpreter, or the OS itself is suspect, as Ken Thompson, one of the original developers of UNIX, pointed out in his Turing award lecture [3]. As a demonstration of this idea, Ken Thompson suggested building a compiler that would take source code of a legitimate program and compile it, adding a backdoor. If someone attempted to replace the malicious compiler, he would have to compile the new compiler using the malicious compiler, and the malicious compiler would turn the new compiler malicious as well.

The above considerations suggest that the question of whether a program can be trusted can not be answered lightly.

4.3 Beyond Code Examination

Further, we believe that code examination alone does not by itself assure the trustworthiness of *an output presented as evidence*. At least the following additional conditions must be assured:

1. the correctness of external inputs of interest at the time of the output's generation, such as the *wall clock time* needed to establish the events' timeline, can be attested, in particular,
2. the configuration of the platform, the operating systems, and of supporting software can be attested, and
3. the specific version of the code must be linkable with the particular evidentiary output.

We believe that the platform on which such software is to be run must be able to **attest** the above properties. In particular, it must be able to attest the platform's configuration at the time of the evidence generation, as well as measure the running version of the software at that time.

5 The Need for Attesting Configuration

In this section we discuss the two fundamental challenges of ensuring the trustworthiness of the evidence-generating system *at the time when autonomous software generates evidence*, a point important for legal analysis. We argue that on a Trusted Computing platform, this issue can and should be addressed by *attestation of the system's configuration.*

Broadly speaking, such attestation is necessary to argue that the channels over which the system receives external, trustworthiness-critical inputs are themselves not compromised or misconfigured, and are not a source of errors introduced into the software's operation.

5.1 Ensuring Accurate Wall Clock Time

An accurate *timeline* is critical to many kinds of both criminal and civil cases. Prosecution's versions of the timeline are routinely contested by the defense. Forensic specialists, in particular, are advised to keep accurate, timed records of their activities.

Computer-generated evidence will almost certainly contain timestamps; in the Roy case, the timestamp of the alleged filesharing activities was one of the two principal elements of the subpoena that directed the ISP to name the defendant and thus subject her to a considerable invasion of privacy and other hardships.

Thus, a natural question to ask is, **"How trustworthy are computer-generated timestamps?"** The answer is common wisdom among computer scientists: not very trustworthy, *unless* either a rigorous clock synchronization mechanism is in place or the system has the benefit of a high-precision external clock (which may synchronize with the true wall clock time by its own means such as GPS or the atomic clock time signal).

It should be noted that when – as it was in the Roy case – electronic evidence involves correlation of events by **two** clock readings (e.g., that of the evidence-generating software/platform and of an ISP's DHCP log server), **both** clocks should be held to the same trustworthiness standards. In this article we concentrate on the requirements to the former, but it should be understood that the latter may also be the source of disastrous timeline errors. For example innocent customers' homes have been reportedly raided by the police due to an ISP's timestamp handling errors "blamed on confusion ... over international time zones"[13], mostly likely due to a software error.[14]

Clock synchronization is a research problem. Clock time synchronization in computers across networks is an important practical and research problem and should not be taken for granted. Dedicated network protocols such as the Network Time Protocol (NTP)[15] are used to synchronize computer system time with dedicated *time servers* trusted to have the accurate time (maintained, e.g., by the US NIST). Network security professionals stress the importance of correct network time synchronization.[16]

The problem of time synchronization is far from trivial. An MIT's Media Lab 1999 survey of NTP network time servers concluded that "only 28% of the Internet based stratum 1 clocks actually appears to be useful", and over a third had deviations of over 10 seconds, and some deviated by hours, days, and even years.[17]

---

[13] `http://www.theregister.co.uk/2009/07/23/intercept_commisisoner/`

[14] Whereas the article quotes a UK government official as saying that "better checks and balances have been put in place", the fault appears to be with the algorithm or process for handling and correlating the timing data, rather than with actual or potential abuse of power.

[15] `http://www.ntp.org`

[16] `http://www.linuxdevcenter.com/pub/a/linux/2003/01/02/ntp.html`

[17] Nelson Minar, *A Survey of the NTP Network*, `http://www.eecis.udel.edu/~mills/database/reports/ntp-survey99-minar.pdf`

Even though network time keeping practices have improved over the years, the issue still attracts attention of researchers and practitioners: the original survey quoted above was followed by at least five since.[18]

Trustworthiness of timestamps must be attested. The above considerations suggest that special steps must be taken to assure the correctness of timestamps on a platform where an evidence-producing software runs, at the time it runs. Since commodity platforms possess neither high quality clocks nor built-in means of synchronization with superior clocks, the actual source of correct time for a commodity platform must be *external*.

This external clock can be either a directly connected device, or a network-accessible time authority (e.g., via NTP). In either case, **the means of synchronization must be configured as a part of the OS configuration process**, and **the configuration active at the point of evidence generation must be attested.**

These requirements, which become self-evident after the above consideration, can be viewed as a design guideline for Trusted Computing platforms and software stacks, one that these architectures should be well-equipped to handle.

5.2 Ensuring Correct View of the Network

Whenever software-generated evidence involves data derived from its network connections – be it the primary subject-matter of its reports, or simply its NTP functionality – the trustworthiness of a system running this software crucially depends on the correctness of its network configuration.

This can be seen from the fact that mapping out and compromising the target systems' *trust relationships* is the methodological foundation of network security assessment and penetration testing (and constitutes core functionality of classic network security tools as *Nmap*, *Nessus*, and *Core Impact*). Moreover, man-in-the-middle attacks on these relationship are the mainstay of attack trees and the reason why vulnerabilities in protocols used to establish network trust such as DNS attract great attention and scrutiny among computer security practitioners.[19]

In the Roy evidence, the evidence-generating system apparently attempted to test the network path taken by the packets, by performing a standard "traceroute" action. However, the results shown in Figure 7 cannot be considered realistic – they neither contain any IP addresses or host names of intermediate hops, nor show realistic hop timings even if we assume that the per-hop tests were actually performed, as it is entirely unrealistic to expect uniform 20ms times on each hop.

This raises the question of whether other actions of the software suffered from whatever caused the apparent failure of the route tracing. This illustrates our point that **full, attested network configuration** is necessary for judging the evidence-generating system's trustworthiness.

[18] See, e.g., `http://www.ntpsurvey.arauc.br/globecom-ntp-paper.pdf`

[19] E.g., Dan Kaminsky's report of a vulnerability in DNS at BlackHat 2007.

6 Conclusion and Challenges

Even though software-generated evidence tends to be regarded as inherently trustworthy by courts, we argue that a number of hard technical problems must be solved in order for such evidence to actually become trustworthy. We believe that the research community must rise to the challenge presented by these inter-related technical, legal, and sociological issues, and develop the – currently lacking – trustworthiness criteria based on the state-of-the-art trustworthy computing approaches.

Part II: Software and Hardware as Witnesses in Trial

7 The Law's Approach to Machines, Software, and Their Reports as Witnesses

A constitutional, country-wide, specific rule has yet to be clearly established in the United States on the issue of the admissibility of, reliability of, and cross-examination of the validity of the underlying theory or algorithm contained in software used as evidence, the machine used to create a report, the source code used on the machine, or the humans operating, maintaining, and otherwise in contact with the machine and source code. However, it can be concluded that, by and large, defendants in the United States will have to demonstrate their need to obtain pre-trial records and testimony on these people, things, and topics and may bear the initial burden in challenging their admission into evidence at trial. A review of cases admitting evidence and expert testimony based on evidence reveals that distrust of the machines used to create evidence and the software running on these machines is a fairly rare commodity, despite technical challenges to accuracy of such machines and their source code.

Defendants in criminal cases benefit from rights under the Bill of Rights of the U.S. Constitution, including, relevant to this discussion, the Sixth Amendment right to confront witnesses against them, known as the Confrontation Clause. The Sixth Amendment provides in relevant part as follows: "In all criminal prosecutions, the accused shall enjoy the right . . . to be confronted with the witnesses against him." U.S. Const., Amend. VI. This constitutional right is available whether the defendant in a criminal case is in state or federal court[20]. The Confrontation Clause - which requires the production of the witness against a defendant at the trial on the criminal matter so that that witness may be cross-examined – represents one of many ways to test the reliability of evidence, but it is the only method guaranteed to defendants. In short, if the Confrontation Clause is implicated, the defendant's task in challenging the evidence is made easier than if the defendant must rely on the rules of evidence, discussed infra.

[20] See Melendez-Diaz v. Massachusetts, ___ U.S. ___, 129 S.Ct. 2527 (2009) (citing Pointer v. Texas, 380 U.S. 400, 403 (1965) for the proposition that the Sixth Amendment is applicable to the States via the Fourteenth Amendment).

If the Confrontation Clause is triggered, the prosecution must produce at trial[21] the witness who made the out-of-court statement so that that witness may be cross-examined by the defendant. Failure to do so renders the out-of-court statement inadmissible.[22] The Crawford case provides an example of exclusion of an out-of-court statement. Michael Crawford stabbed a man named Kenneth Lee who allegedly tried to rape Mr. Crawford's wife Sylvia earlier that night. Michael Crawford was convicted of assault with a deadly weapon after the prosecution played for the jury a tape-recorded statement by Mrs. Crawford, recorded immediately after the incident during police interrogation, which discredited Mr. Crawford's argument that he acted in self-defense. Before the case came to the U.S. Supreme Court, the Washington Supreme Court upheld Mr. Crawford's conviction and had determined that the recorded statement was reliable.[23] Due to the marital privilege, Mrs. Crawford was unavailable to testify at trial and unavailable to be cross-examined by the defendant outside of trial. The U.S. Supreme Court held that the recorded statement of Mrs. Crawford, made out-of-court, should not have been admitted as evidence since it was a testimonial statement[24] and Mr. Crawford would not have an opportunity to cross-examine Mrs. Crawford during trial, in violation of his constitutional right under the Confrontation Clause.

Citing an 1828 dictionary to bolster the U.S. Constitution's framer' intent in light of a series of English cases, U.S. Supreme Court Justice Antonin Scalia, writing the majority opinion in Crawford v. Washington, 541 U.S. 36 (2004), equated the meaning of "witnesses" to be those who "bear testimony."[25] Triggering of the Confrontation Clause is determined based on whether an out-of-court statement is testimonial or non-testimonial. "Testimony" was defined in the 1828 dictionary as follows: "'[a] solemn declaration or affirmation made for the purpose of establishing or proving some fact.' "[26]

Being a human being providing a statement during police interrogation, Sylvia Crawford was easy to identify as a witness bearing testimony, generating a constitutional requirement that she, in essence, be produced as a witness at trial. Whether that constitutional requirement applies to machines, operators of

---

[21] If the prosecution cannot produce at trial the testimonial witness against the defendant, the out-of-court statement by that witness is inadmissible unless - generally speaking – the prosecution establishes that the witness is unavailable to testify and the defendant has had an opportunity to cross-examine the witness. Crawford v. Washington, 541 U.S. 36, 5457 (2004).

[22] See prior footnote.

[23] State v. Crawford, 54 P.3d 656, 663 (2002) (overturning the lower court's decision that the statement was unreliable, State v. Crawford, 107 Wash.App. 1025 (Wash.App. Div. 2 2001)).

[24] The out-of-court testimonial statement may be either a sworn document or unsworn and will still invoke the Confrontation Clause requirement. Crawford v. Washington, 541 U.S. 36, 52, n.3 (2004).

[25] Crawford v. Washington, 541 U.S. 36, 51 (2004) (citing 2 N. Webster, An American Dictionary of the English Language (1828)).

[26] Id.

machines, and/or the makers of the machines and their source code remains an open question.

The recent U.S. Supreme Court case Melendez-Diaz v. Massachusetts, the majority opinion for which was also written by Justice Scalia, held that the analyst in the state forensic crime lab who provided a certificate that a particular substance was cocaine must be brought to trial by prosecutors (to enable cross-examination by the defendant) in order to render his certificate of the substance admissible. Under prior law, such lab technician certificates were considered reliable, and therefore not subject to the method of testing that reliability found in the Confrontation Clause, namely, cross-examination of the human signing the certificate presumably reporting results following the use of lab equipment that need be calibrated, operated correctly, possibly with a series of repeated tests, and possibly confirmed with a alternative test method reflecting an alternative underlying principle or algorithm to test for cocaine. See id. at 2537-38 (speculation as to the lab technician's method and techniques in reaching the reported conclusion). The analyst's actions, choice of equipment and tests to use, and methodology were unknown in this case, because none of that information had been admitted; merely the certificate stating the substance was cocaine was admitted in the trial court.

The Melendez-Diaz court eschewed the contrary views that reliability of such lab results need not be tested. Citing a 2009 report prepared by the National Academy of Sciences for a number of error-provoking factors present with such testing, the court found that "[f]orensic evidence is not uniquely immune from the risk of manipulation." Id. at 2536 (citing National Research Council of the National Academies, Strengthening Forensic Science in the United States: A Path Forward (Prepublication Copy Feb. 2009)). Some of the findings of the report were that labs are not neutral, but administered by law enforcement personnel, providing incentive to alter evidence and that the "[f]orensic science system . . . has serious problems." The latter problem involves lack of competency or failure to exercise sound judgment by the analyst. The court cited a study of wrongful, overturned convictions which "concluded that invalid forensic testimony contributed to convictions in 60% of cases." Id. at 2537 (citing Garrett & Neufeld, Invalid Forensic Science Testimony and Wrongful Convictions, 95 Va. L.Rev. 1, 14 (2009)). The court also cited the National Academy of Sciences report for the proposition that, among other information crucial to creating reliable results, sound methodologies in published material are lacking across forensic science disciplines, resulting in, among other problems, unreliability in even commonly used forensic tests such as fingerprinting and firearms analysis. Id. at 2538. The National Academy of Sciences report suggests that the development of a sound methodology would require published material leading to a general acceptability of the methodology, with published material available to analysis with regard to techniques, research, and types and numbers of potential errors. Id.

In the case U.S. v. Washington, 498 F.3d 225 (4th Cir. 2007), cert. den'd., 129 S. Ct. 2856 (2009[27]), the Fourth Circuit declined to determine that data generated by a lab machine was testimonial. The machinery consisted of a Hewlett Packard chromatograph and a computer using Hewlett Packard ChemStation software. The Fourth Circuit upheld the prosecution's presentation at trial of the supervising director of the lab to interpret the machine's data report and neither the three lab technicians who used the machinery and software nor the machines themselves (not discussing the possibility to cross-examine Hewlett Packard's software engineers). The court pointed out that the Confrontation Clause's cross-examination requirement applies to "(human) 'witnesses."' Id. at 230, n.1.

There appears to be no right under the Confrontation Clause for a defendant in a criminal case to cross-examine the software developers or machine designers. See, e.g. U.S. v. Washington, discussed supra (and cases cited therein holding, respectively, that time stamp on fax print out, header on print out of Internet images, and computerized telephone trace report are not testimonial statements); see also State v. Chun, 943 A.2d 114, 148 (NJ 2008) (determining that the print-out from a breath alcohol measurement device and associated software and hardware is not a testimonial statement). However, recent U.S. Supreme Court precedent acknowledges defendants' constitutional right to cross-examine analysts using devices and software. It is unclear whether such a right can be extended to software programmers and, if so, under what circumstances.

Once the realm of constitutional protections is left, the burdens on the defendant to find evidence bearing on the reliability of the evidence increase. For example, while a prosecutor bears the burden to prove the chain of custody for evidence, he need not prove every step in the chain of custody, and any lacking evidence merely goes to the weight that may be given to the evidence, not to the admissibility of the evidence.

One of the gatekeeping tools available to defendants to prevent unreliable evidence from becoming admissible is the hearsay rule found in the rules of evidence.[28] "'Hearsay' is a statement, other than one made by the declarant while

[27] The U.S. Supreme Court denied certiorari, declining to review the Fourth Circuit's decision in this case, four days after it issued its opinion in Melendez-Diaz v. Massachusetts, ___ U.S. ___, 129 S. Ct. 2527 (2009).

[28] Each court has its own rules governing admissibility of evidence into trial. Federal courts follow the Federal Rules of Evidence and state courts are free to adopt their own rules of evidence. Generally speaking, however, state rules of evidence closely follow the Federal Rules of Evidence. Much of the discussion about admissibility of evidence revolves around the interpretation of the rules of evidence, in particular, interpretation of the hearsay rule and expert testimony rule. Although state courts may use federal decisions interpreting the Federal Rules of Evidence as guidance for interpreting their state rules of evidence, see, e.g., N.H. Rules of Ev., Rule 102, their decision - except where rights granted by the U.S. Constitution control - need not follow the federal decision. See, e.g., Alice B. Lustre, J.D., Post-Daubert Standards for Admissibility of Scientific and Other Expert Evidence in State Courts, 90 ALR5th 453 (2001).

testifying at the trial or hearing, offered in evidence to prove the truth of the matter asserted." Fed. R. of Ev. Rule 801(c). "A 'declarant' is a person . . ." Fed. R. of Ev. Rule 801(b). Generally speaking, hearsay is inadmissible. Fed. R. of Ev. Rule 802. Excepted from the hearsay rule are records made in the regular course of business, Rule 803(6) and reports prepared by public offices pursuant to a duty to so report except for law enforcement personnel reports for criminal cases, Rule 803(8). Also excepted from the hearsay rule are statements containing material facts which may not otherwise be procured through reasonable efforts and have equivalent circumstantial guarantees of trustworthiness. Rule 807. Data reports - so long as they may not be considered "testimonial" – are often sought to be admitted into evidence under the business records exception (and sometimes other exceptions) to the hearsay rule. State v. Chun, 943 A.2d 114, 166 (NJ 2008) (also indicating that machines do not have an intent to generate a false positive); see Thomas v. U.S., 914 A.2d 1, 13 (D.C. 2006); see also Crawford v. Washington, 541 U.S. 36 (2004).

Given the possible interplay between the Confrontation Clause and the business records exception to the hearsay rule, prosecutors should separate testimonial (i.e. a solemn affirmation made for the purpose of establishing or proving a fact) statements, which are inadmissible in criminal cases without the defendant's ability to cross-examine the witness, from data logs prepared in the ordinary course of business which are designed to render a conclusion at the time the data is generated. A continuum of types of reports and logs could be envisioned which leads to a difficulty to deciding at what point a piece of evidence is a data log and at what point it is a testimonial statement. This struggle is apparent in the cases, and yet a review of the cases fails to illuminate where the line dividing the two will gel.

Also, data logs prepared in anticipation or in preparation for litigation generally do not fall within an exception to the hearsay rule, see Thomas v. U.S., 914 A.2d 1, 13 (D.C. 2006). Consequently, defendants may subpoena witnesses to testify on the facts surrounding production of the data logs so long as the defendant establishes a lack of reliability justifying the subpoena. See State v. Chun, 943 A.2d 114, 166 (NJ 2008). This shift of burden can be difficult to overcome. See UMG Recordings, Inc. v. Lindor, U.S. Dist. Court, E.D.N.Y., Docket No. 05-CV-1095 (May 16, 2008 Order preliminarily denying defendant's motion to compel production of source code by MediaSentry in allegedly detecting allegedly unlawful copyright infringement).

In the area of source code and hardware design matters, defendants bear the additional difficulty of needing to overcome the creator's allegations that the code/design is proprietary and consequent unwillingness to produce the code/design. In both State v. Chun and UMG v. Lindor, the source code developer initially fought discovery of source code due to the allegedly proprietary nature of the code. In Chun, a case involving source code used in Alcotest, a device and software used to detect blood alcohol level for use in driving while intoxicated cases, the German code developer did produce the code, which was subsequently evaluated by defendants' experts, resulting eventually in a require-

ment to modify the code to correct errors. By contrast, the software code used to allegedly detect and allegedly produce accurate screen shots of the defendant's computer was not produced in the Lindor case. Even if it had been, the code involved would have been subject to a confidentiality restriction, such that evaluation of the code had to occur on an (expensive, time-consuming, and inefficient) defendant-by-defendant, case-by-case basis. Unlike the code evaluation conducted by experts in the public eye in the Chun case, the code used in the Lindor case - if the court had compelled its discovery, which it did not – could not be tested for reliability in such a way that subsequent defendants could use it.

When determining whether expert testimony is admissible, reliability of the methodology used by the expert is crucial to the decision. See Daubert v. Merrell Dow Pharmaceuticals, Inc., 509 U.S. 579 (1993). One of the key ways to determine whether a new technology's methodology is reliable is whether it is generally accepted after an opportunity for peer review and has reliable results. See id. If no duplication of the forensic testing by another methodology is possible (such as is the case in the alleged detection of peer-to-peer network sharing of copyrightable works, which occurs at a specific instant in time and in the case of deterioration of samples, such as blood alcohol content samples and autopsies (which may not be repeated)[29]), reliability of methodology is difficult to determine.

Despite these concerns, some courts have admitted computer forensic evidence. See, e.g., UMG Recordings, Inc. v. Lindor, 531 F.Supp.2d 453 (2007) (admitting opinion of plaintiff's expert on facts bearing on copyright infringement claim despite failure of the methodology to comport with Daubert factors in light of expert's own testimony that others in the industry would interpret the data the same and court's conclusion that data relied upon by expert was "objective data" provided by plaintiffs' private investigator and ISP records); Galaxy Computer Servs., Inc. v. Baker, 325 B.R. 544 (E.D. Va. 2005) (admitting expert testimony that former officers of corporation deleted files from computer after conspiracy and other claims were brought against them following specially-educated and seasoned computer forensic specialist's analysis of hard drives); see also Marjorie A. Shields, J.D., Admissibility of Computer Forensic Testimony, 40 ALR6th 355 (2008) (describing eight cases where the computer forensic testimony was admitted and only one where the testimony was not admitted; in the case where the testimony was not admitted, the alleged expert was unable to even open the AVI files that he was supposedly hired to opine did not exist and were not pornographic (this inability to open the files following his initial inability to locate the files on the computer)). This suggests that in practice there is a low threshold for computer forensic evidence, which places significant burdens on defendants to challenge reliability of this evidence. See also David L. Faigman, David H. Kaye, Michael J. Saks, Joseph Sanders, 5 Modern Scientific Evidence: The Law and Science of Expert Testimony 41:13 (Nov. 2009) (citing State v. Bastos, 985 So.2d 37 (Fla. Dist. Ct. App. 3d Dist. 2008), in which the court refused to order a turnover of source code absent a

[29] Crawford, at n.5.

particularized showing of discrepancy; People v. Robinson, 53 A.D.3d 63, 860 N.Y.S.2d 159 (2d Dept. 2008) (similar); State v. Underdahl, 749 N.W.2d 117 (Minn. Ct. App. 2008) (similar)); but see State v. Chun, supra (allowing thorough evaluation of source code); but see House v. Com, 2008 WL 162212 (Ky. Ct. App. 2008) (ordering disclosure of source code).

In conclusion, while the recent expansion of rights to defendants in criminal cases to require prosecutors to bring lab analysts into court for cross-examination and to produce documents establishing the proper calibration of machines and training of operators of machines is a positive step in the testing of reliability of computer-aided forensic evidence and resulting expert testimony, these rights have yet to gain much benefit for defendants in civil cases faced with the admissibility of evidence and expert opinion that very possibly lack peer-tested methodologies, trustworthiness, and/or competency. A survey of civil court cases suggests a lenience toward admitting evidence and opinions and allowing the jury to sort out the weight to be afforded, which can unfavorable prejudice defendants in civil cases. When a defendant in a civil case can end up with a verdict of $1.92 million[30] for sharing 24 copyrighted songs on a peer-to-peer network, civil cases begin to look as if they should require the reliability and confrontation standards available to defendants in criminal cases.

8 Acknowledgements

The first and the third author would like to thank Sean Smith, Denise Anthony, and Thomas Candon, who encouraged our interest in social aspects of security. We are also grateful to our colleagues at Dartmouth's PKI/Trust Lab and the Institute for Security, Technology, and Society for their support.

The second author would like to acknowledge the support of her colleagues and the administration at Franklin Pierce Law Center.

We would like to thank Ray Beckerman and anonymous TRUST reviewers for helpful comments.

References

1. J. Bannet, D.W. Price, A. Rudys, J. Singer, and D.S. Wallach. Hack-a-vote: Demonstrating security issues with electronic voting systems. *IEEE Security and Privacy Magazine*, 2(1):32–37, 2004.
2. G.J. Holzmann et al. The model checker SPIN. *IEEE Transactions on software engineering*, 23(5):279–295, 1997.
3. K. Thompson. Reflections on trusting trust. In *ACM Turing award lectures*, page 1983. ACM, 2007.

[30] "Music Labels Win $2 Million in Web Case," New York Times (June 18, 2009). The verdict was later reduced to $54,000. "Judge slashes "monstrous" P2P award by 97% to $54,000", Nate Anderson, Ars Technica (January 22, 2010), `http://arstechnica.com/tech-policy/news/2010/01/judge-slashes-monstrous-jammie-thomas-p2p-award-by-35x.ars`.