Data provenance with watermarks for usage control monitors at disaster recovery

Martin Salfer, Sven Wohlgemuth, Sebastian Schrittwieser, Bernhard Bauer, Isao Echizen

Angaben zur Veröffentlichung / Publication details:

Salfer, Martin, Sven Wohlgemuth, Sebastian Schrittwieser, Bernhard Bauer, and Isao Echizen. 2011. "Data provenance with watermarks for usage control monitors at disaster recovery." In *Proceedings of the 2011 International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing, iThings/CPSCom 2011, 19-22 October 2011, Dalian, China*, edited by Feng Xia, Zhikui Chen, Gang Pan, Laurence T. Yang, and Jianhua Ma, 514–19. Los Alamitos, CA: IEEE. https://doi.org/10.1109/ithings/cpscom.2011.129.

Data Provenance with Watermarks for Usage Control Monitors at Disaster Recovery

Martin Salfer, Sven Wohlgemuth, Sebastian Schrittwieser, Bernhard Bauer, Isao Echizen

National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku Tokyo 101-8430, Japan
{salfer,wohlgemuth,iechizen}@nii.ac.jp
TU Vienna, Favoritenstrasse 9-11/188, 1040 Vienna, Austria
sebastian.schrittwieser@tuwien.ac.at
Universität Augsburg, Universitätsstr. 2, 86159 Augsburg, Germany
bernhard.bauer@informatik.uni-augsburg.de

***Abstract*—Security restrictions are often lifted for faster disaster recovery. This grants access to external services for providing at least basic infrastructure operation, but puts data at risk. Usage control monitors are supposed to avoid data abuse by watching every movement, yet they appear unfit for a fast roll out and prone to low-level attacks and steganographic tunneling. We propose supplementing monitors with watermarkers and also sketch out an algorithm for it to integrate provenance information into data itself. So, an auditor can inspect such watermarks, query involved monitors, and tell a legitimate from an illegitimate usage more easily.**

***Keywords*-Monitoring; Watermarking; Tunneling; Data Provenance; Resilience**

I. Introduction

The United Nations predict for 2050 that the concentration of the world's population living in cities will increase from 50% to 70% [1]. ICT infrastructures, which urban people greatly depend on, can be disturbed by crime, terrorism, or disaster. Hence, protection standards and guidelines demand deployment of redundant and security certified systems and even take external services into consideration as long as they are certified and trusted well enough. Yet, this requirement only holds as long as expected threats become reality.

A system should be resilient, which is being able to handle even unexpected disasters, stay in an equilibrium and eventually recover. The ability to fast integrate external services heightens flexibility and resilience, and many IT techniques, e.g., cloud computing and service-oriented architectures (SOA), support this. Yet, integration might still be denied due to security concerns, e.g., the Japanese SINET denied integration of commercial services during the 2011 Tohoku earthquake[1] due to security reasons.

Monitoring is considered a key aspect for ensuring data compliance [3] and could also facilitate secure integrating of external services by logging, checking, and even mediating every data access and usage. This eases abuse investigations and therefore makes committing crimes less attractive and less likely. Yet, a presumption for a successful tracking and enforcement is sensing all relevant events and data flows of ICT infrastructures.

As a contribution:

- We surveyed selected usage control monitors in Section II for resilience fitness and identified neglect of lower levels and backtracking as a commonality.
- We construct two attacker models in Section III: an insider attacking from low level, and an outsider remotely injecting an exploit carrying a steganographic engine to tunnel data through a network covertly.
- We propose an algorithm idea in Section IV for enhancing monitors with a watermarking component that inserts data provenance[2] information for backtracking leaked data – even if the data was tunneled.

II. Selected Monitoring Proposals

Enforcing data policies and data protection laws is based on monitoring data transfers, keeping logs of the relevant information, and, in case of a violation, take appropriate action. Here, we briefly examine a few usage control monitoring approaches that use ad hoc monitoring and interception and also some that audit events ex post.

A. Centralized Rewriting Monitoring

The following monitors are usually centrally controlled and apply re-writing mechanisms. A common analog to those monitors are firewalls: Both examine, rewrite requests and keep logs about events and its data.

The *Metric First-Order Temporal Logic based distributed systems monitor by Basin et al.* [5] is a system extension for logging events and checking policy violations with temporal logic specifications. The focus is on distributed systems, where merging and ordering log events is an intractable problem. They accomplish specification of several basic

[1] At the Tohoku 2011 earthquake, also referred to as the Great East Japan Earthquake of March 11, 2011, several private telecommunication infrastructures broke down, but the Japanese science information network (SINET) remained available, even in Sendai [2]. Those intact ICT infrastructures could have supported the damaged ones, yet they were kept isolated due to IT security concerns.

[2] "Data provenance – sometimes called "lineage" or "pedigree" – is the description of the origins of a piece of data and the process by which it arrived in a database" [4].

rules and also an implementation of a satisfyingly fast monitor, but also admit that policy checking correctness relies on complete monitoring of events, which they could not reach but plan to compensate with future audit techniques that tolerate incomplete logs. Hence, their approach greatly heightens data compliance for a trusted and established system, but seems not to consider backtracking data that moved covertly.

MASTER by EU's Seventh Framework Programme (FP7) [6] aims at achieving compliance for integrating external services into SOA and addresses new challenges like distributed ownership and cross domain operations. They outline a methodology and high-level architecture including a monitoring infrastructure for observing key indicators, and reaction components for auto enforcement and audits by human users in case automated policies are in doubt.

The "Providence" framework by Khazankin and Dustdar [7] intercepts messages and searches in those for sensitive data. It assumes that all existing sensitive data is being registered in the monitors database.

The *model-driven event monitoring rules automation proposal by Giblin et al.* [3] is for "monitoring of application events to determine whether business processes and applications operate within the parameters set forth in formal compliance policies." They utilize event reporting functionalities of IBM applications as sensors and evaluate with model-driven, auto-generated monitor policies. It's main strength is the automatic roll out of policy rules onto new monitors.

The *obligation monitor prototype by Hilty et al.* [8] proposes monitors especially also on the consumer side. Unfortunately, user side signaling might be unreliable and not reach the provider. Queueing signals is a proposed solution for this problem.

The *SMTP Proxy by Takebayashi et al.* [9] is a monitor that is installed between email clients and servers for maximum compatibility. Its goal is to avoid mistakes when sending email data, which according to them happen most often "during the lunch break and at the end of the working day". Beneficial for a disaster application, they even consider malicious information disclosure and the use of text recognition to handle document citations or slight text modifications[3].

B. Ex Post Usage Control with Watermarks

The cited approaches watermark data with data provenance information, but do not apply rewriting mechanisms. So, unintended usage is possible, but auditors can investigate on data provenance information at a later point in time and then take action accordingly.

The *watermarking tests for the European Broadcasting Union (EBU) by Cheveau et al.* [10] evaluated four commercial-grade watermarkers for backtracking leaked multimedia data. Their aim is to secretly embed intellectual rights and broadcasting information into data payload. One requirement was cascading at least three watermarks, i.e., a watermarker must be able to add a watermark into already watermarked data without destroying existing watermarks. As their approach uses watermarks for storing data provenance information, it seems fit for disasters that destroy logs, but their approach is basically designed for data distribution over merely three nodes, producer, distributer, and end consumer, and assumes a fixed hierarchy of these, which simplifies key management.

DETECTIVE by Wohlgemuth et al. [11] is deriving data provenance by watermarks and checking obligations for validating usage control ex post. As it uses watermarks, logs are not necessary either, and unauthorized data disclosure are backtrackable even if data is moved secretly to somewhere else. DETECTIVE provides higher cryptographic protocols that require data owners to participate. This approach seems useful for exchanging sensitive data between cooperating services, but requires a direct integration into a service for taking part in the cryptographic protocol and can only backtrack participant wise but not finer.

C. Monitoring Supplements

The cited contributions convey some beneficial features for monitoring systems.

BBox by R. Accorsi [12] is a "digital black box to provide for authentic archiving in distributed systems" and is supporting compliance audits with reliable logs. It cryptographically secures log message storage and transmission between sources, collector, and auditors and furthermore even is able to disclose records to auditors selectively. Even though BBox's understood limit is "check[ing] the veracity of the [logged] events", whether log statements are factually true, BBox explicitly demands trusted computing and can guarantee the log authenticity as only cryptographically authenticated devices can submit log messages for supporting ex post auditors.

"Usage Control Enforcement" by Pretschner et al. [13] points out that "most current control mechanisms [...] are concerned with DRM of multimedia data[,] are implemented in software and located at the application layer." Within the survey, the High Bandwidth Digital Content Protection (HDCP) stood out as being able to degrade data quality for unauthorized use, which seems to make acceptance and integration easier.

D. Summary

Those monitors offer sophisticated data usage policy modeling and checking techniques, yet they are dependent onto the extend of their logging mechanisms. They can only

[3]Their SMTP proxy matches text structure patterns as a hashing could be tricked too easily by doing modifications or citations.

detect what they were specified for, yet steganography[4] continually finds new ways into covert channels and therefore trick existing monitors, demonstrating their limitations.

Low-Level attacks are often neglected even though they are generally applicable and subversive. For example, an operating system driver can provide monitors with fake data or steal data. As most monitors are located at the application layer, an attacker merely needs to exploit an application library or its operating system kernel. Only one paper out of the above examined ones (BBox [12]) explicitly mentions that threat and demanded a hardened or trusted platform. Although these can not avoid low-level attacks, they lower its probability. Thus monitoring concepts are urged to take subverted entities into account.

Most monitoring concepts lack backtracking features. If data is found somewhere it is not supposed to be, it will be hard to tell where it came from as all instances of some data are identically to each other. Watermarking approaches generate many different versions of a data package, making approximate backtracking possible.

III. Undermining Monitoring

Current literature often fails to take low-level attacks and steganographic tunneling into consideration. We outline two attacker models using these (as visualized in Figure 1 and briefly show why tunneling is generally undetectable.

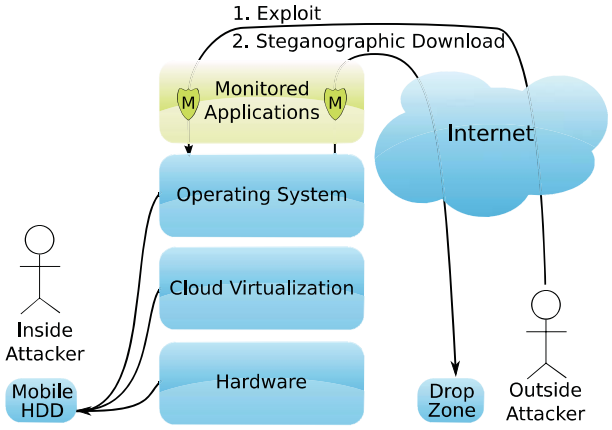

Figure 1. Insider and Outsider Attacker Model.

A. Insider Attacker Model

An inside attacker can easily commit manipulations as he or she usually has full access to all implementation levels: from the application level down to the physical level.

First, he or she would install an abstraction layer directly under the lowest monitored layer. As most monitors run on the application level, root access to the operating system kernel is sufficient for getting undetectable access. In case a monitor is on the operating system level, an attacker can let the system run on an (universally applicable) hypervisor[5], and in case all software is monitored, he or she can directly tamper with hardware to get data access.

Second, the attacker can scan all IO transactions and memory contents for relevant data.

Finally, he or she can download data and carry it away in removable mass storage devices, e.g., USB flash drives.

B. Outsider Attacker Model

An outside attacker can send in a remote exploit for a low-level system including an engine for setting up a steganographic tunnel. Such tunnels usually offer only a few percent of bandwidth, but even a 2.5 kbps steganography tunnel over VoIP [16] is enough to steal a large amount of data[6], and even high-bandwidth is achievable in combination with encryption as seen with the anti-censorship proof of concept Telex from Wustrow et. al [17] that can transfer data covertly with an average overhead of only 60%. It is a newly proposed combination for tricking monitors. It sends out steganographically enhanced Transport Layer Security (TLS) segments and encrypts payload also with TLS to innocent looking destinations that are eventually rerouted to the destination that is given in the initial steganographically hidden commands. This attacker model is even possible with a 100% loyal and honest service provider and is especially likely during disasters where security models are lifted or loosened, enlarging attack surface.

C. General Undetectability of Steganographic Tunnels

Steganographers continually find new ways of tunneling data as monitors can only detect and disturb known steganographic techniques [18] [19], but a general detection stays an unsolvable Entscheidungsproblem [20] as Rice's theorem [21] says that, "Any nontrivial semantic property of programs is undecidable." [22]. To conform to this theorem:

- The *"semantic property"* of containing steganographic data has to be expressed as a class of languages, in our case "the languages (for packages) that contain a steganographic payload",

$$\mathcal{S}_{\text{Payloaded}} = \{L \mid L \text{ uses a steganographic payload}\}$$

[4]Steganography is the art of embedding data secretly into other data. It arises less suspicion than encryption as it looks more innocent. Data to be embed is called the *payload*, and data to carry the payload is called the *carrier*. The result of encoding a payload into a carrier is called the *package*.

[5]A Hypervisor is an optional virtualization layer between an operating system (OS) and hardware and is usually used for running several OSs on one machine simultaneously. It can be installed even on the fly and is hardly noticeable for upper levels as it can manipulate system calls accordingly. Rutkowska publicized a proof of concept malware hypervisor as "Blue Pill" [14] [15].

[6]That is enough bandwidth for secret documents, e.g., the "Afghan War Diary" compromises about 90 000 secret military documents and fits only 75 MB, which could be transferred within less than three days.

- The *"non-trivial"*requirement has to be fulfilled and can be seen as some Turing recognizable languages are a member of $\mathcal{S}_{Payloaded}$ and some are not, i.e., $\{\langle M\rangle \mid L(M) \in \mathcal{S}_{Payloaded}\} \neq \emptyset \neq \{\langle M\rangle \mid \mathcal{S}\backslash\mathcal{S}_{Payloaded}\}$,
- *"Program"* has to be interpreted as data traffic passing by the monitor.

Thus, monitors can never specify a general detection of steganographic tunnels and must be considered leaky. Hopper et al. [23] even constructed a provably secure steganographic protocol.

IV. Watermarking Monitors

Some monitors have already reached an advanced policy checking level. But since they usually have no access to data provenance information, they cannot backtrack data. So, we want to compensate their leakage by providing data together with provenance information, which is technically possible by embedding logs in watermarks of passing by documents as seen in Figure 2.

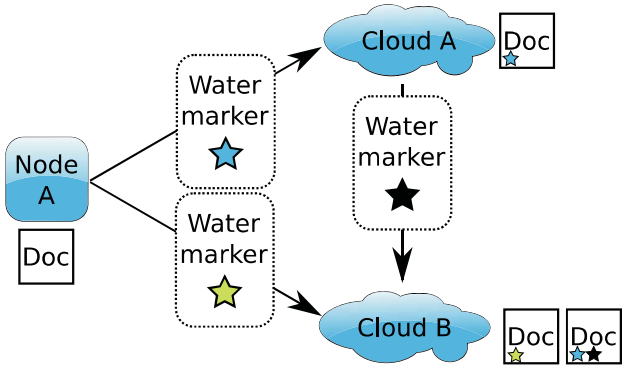

Figure 2. Watermarking Monitors.

A. Incentives for Watermarking Monitors

Advantages come to sight when external services need to be incorporated due to a disaster. Normal usage control monitors usually need time to roll out their protection mechanisms into external infrastructures before data may be released. A thoroughly watermarking system can immediately exchange data as watermarking preserves file formats, creates unique releases. This clarifies responsibility and motivates service provider to keep their systems secure. Responsibility about a data leak can be told when backtracking illegitimate data to its last legitimate owner. The more watermarker are active, the finer backtracking is possible.

B. Recomputing Checksums and Signatures

Data modification usually *invalidates checksums* and *digital signatures*, but a monitor should fix that to avoid disrupting communication. Hence, a monitor must check all checksums and replace those invalid with newly computed ones.

In case a checksum is even backed by a *digital signature*, a monitor can't recompute it without it's secret key. One obvious way is to *provide the monitors with all secret keys* so that they can replace the invalidated signatures with valid ones. But having monitors with access to so many secret keys would violate many security policies, as a secret key is usually supposed to be known to as few as possible.

Another way is *logging all changes and offering a signature checking service*, i.e., whenever a client faces an invalid signature, it can query the monitors to find out whether a watermark was the only alteration. But this is problematic too:

- Centrally stored data can be compromised or become unavailable.
- An interactive service would raise data privacy issues about tracking request behavior, e.g., "who queries, when, where, and for what signature".
- And all affected applications would have to be updated for generating queries.

A more practical way might be *attaching monitor signatures* that certify monitor changes. Someone would still have to implement extensions to enable applications to cope with those extra signatures. But such checks could be carried out independently and repeatedly, making it more failsafe against attacks and faults.

Attaching monitor signatures seems to be a good choice among those three mentioned ones because of its decentralization robustness.

C. Auditing

An auditor can derive data provenance information by inspecting watermarks and querying its creators.

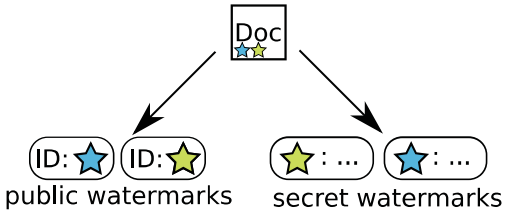

Figure 3. Types of Watermarks in a Document.

The two types of watermarks used in this system can be seen in Figure 3:

- *public watermarks* are readable for everybody and reveal pseudonymous references to monitors that inserted further watermarks into it. These watermarks are supposed to be write protected to avoid covering up tracks.
- *secret watermarks* are read- and writeable and contain data provenance information. These are only accessible for watermark creators, knowing the secret key.

First, an auditor reads the public watermarks and contacts its creators. Then, he or she consult these with the document for receiving embed information (as seen as in Figure 4). Some replies might be wrong as monitors or watermarks might have been compromised, so an auditor should consider setting up some probability model. Finally, the auditor derives data provenance from received, trustful information fragments.

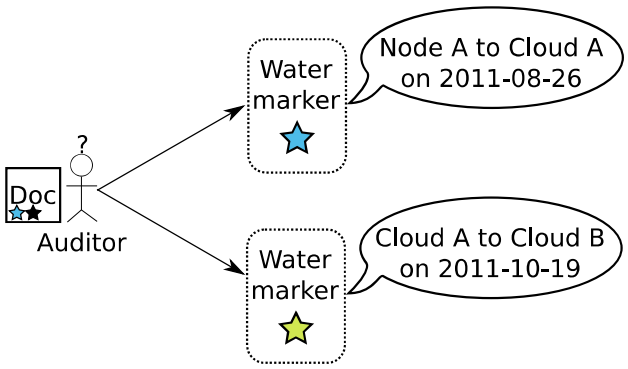

Figure 4. Auditor Queries.

D. Algorithm

A possible algorithm for a watermarker is sketched out:

1) Receive data packet.
2) Validate the legitimacy of the data packet:
   a) Run usual monitoring policy checks.
   b) If necessary request data provenance info:
      i) Extract all monitoring IDs from a data packet's publicly readable watermarks.
      ii) Query watermark creators directly for data provenance information.
   c) Revaluate, and in case escalate to a human.
3) Extract meta information from the packet, e.g., origin and destination.
4) Watermark the data package:
   a) Verify checksums and digital signatures.
   b) Watermark data with the own ID publicly visible.
   c) Watermark data with the above's meta information with a secret key.
   d) Replace invalidated checksums with correct ones.
   e) Attach a digitally signature and for previously correctly signed data using the monitor's secret key.
5) Pass on data packet.

The approach does not contain any obligation checking as the obligation checking is supposed to be done by monitoring systems itself.

E. Challenges

Some challenges watermarking monitors face:

1) The checksum replacement is ineffective when two applications exchange checksums *unnoticed for the monitor*. An API for comparing data might help.
2) Watermarkers might overwrite previously embedded, secret watermarks unintentionally, should look for these and use less interfering algorithms, while auditors should find ways to extrapolate information.
3) Watermarks can only comprise a finite amount of data. Therefore they must handle data capacity economically by using references instead of text descriptions.
4) Monitors should settle on a standard for indicating hidden watermarks that can effectively find and consider previous watermarkers of given data . One proposed tool for that process is keeping watermark metadata [24]. Simmhan et al. [25] mention how important it is to unify efforts onto a meta data standard for data provenance. And watermark based monitoring networks need such standards, too, for creating a more effective, interoperable monitor network.
5) The information in the watermarks and the auditor queries must be reasonably anonymized and comply to privacy laws and regulations. It might even be classified as data retention, which is still fiercely discussed in some jurisdictions.
6) Every additional watermark degrades data quality. Watermarking tests show that especially cascading watermarks that were made with similar algorithms were more disruptive than cascading watermarks that were made with different algorithms [10]. Thus, data *consumers* might have to request data from data *owners* more directly to minimize intermediate watermarkers. And watermarkers might have to use a variety of algorithms.

Solving such challenges and setting up a diverse network of watermarking monitors might yield a transparent and decentralized backtrack system that can even handle destroyed or compromised watermarkers and tunneled data.

V. Conclusion

Some usage controls monitors have already reached advanced policy checking capabilities and might in the near future become as common as firewalls. But both have limitations in detecting intentional data leaks. And that might be partly solved with watermarks as they can backtrack data for monitors. The rise of data storage capacity and communication bandwidth should fuel watermarking because of larger data capacities. A combination of usage control monitors with watermark based data provenance information might yield a more robust security system. One that even withstands most disasters and allows immediate integration of external services for faster disaster recovery.

ACKNOWLEDGEMENT

This work was conducted under the Memorandum of Understanding (MOU) between the National Institute of Informatics (NII), Japan, and the TU Vienna, Austria, and also under the MOU between the NII and the University of Augsburg, Germany.

REFERENCES

[1] United Nations, "An overview of urbanization, internal migration, population distribution and development int the world," http://www.un.org/esa/population/meetings/EGM_PopDist/P01_UNPopDiv.pdf, Jan. 2008.

[2] K. Watanabe, "SINET4 has been offering services, despite the great east japan earthquake," *NII Today*, vol. 38, pp. 8–9, Jul. 2011.

[3] C. Giblin, S. Müller, and B. Pfitzmann, "From regulatory policies to event monitoring rules: Towards Model-Driven compliance automation," Oct. 2006.

[4] J. Van den Bussche, V. Vianu, P. Buneman, S. Khanna, and T. Wang-Chiew, "Why and where: A characterization of data provenance," in *Database Theory — ICDT 2001*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2001, vol. 1973, pp. 316–330.

[5] D. Basin, M. Harvan, F. Klaedtke, and E. Zălinescu, "Monitoring usage-control policies in distributed systems," in *18th International Symposium on Temporal Representation and Reasoning (TIME'11)*, 2011.

[6] V. Lotz, E. Pigout, P. M. Fischer, D. Kossmann, F. Massacci, and A. Pretschner, "Towards systematic achievement of compliance in Service-Oriented architectures: The MASTER approach," *WIRTSCHAFTSINFORMATIK*, vol. 50, no. 5, pp. 383–391, Nov. 2008.

[7] R. Khazankin and S. Dustdar, "Providence: A framework for private data propagation control in service-oriented systems," in *Towards a Service-Based Internet*, ser. Lecture Notes in Computer Science, E. Di Nitto and R. Yahyapour, Eds. Springer Berlin / Heidelberg, 2010, vol. 6481, pp. 76–87, 10.1007/978-3-642-17694-4_7.

[8] M. Hilty, A. Pretschner, D. Basin, C. Schaefer, and T. Walter, "Monitors for usage control," in *Trust Management*, ser. IFIP International Federation for Information Processing, S. Etalle and S. Marsh, Eds. Springer Boston, 2007, vol. 238, pp. 411–414, 10.1007/978-0-387-73655-6_29.

[9] T. Takebayashi, H. Tsuda, T. Hasebe, and R. Masuoka, "Data loss prevention technologies," *Fujitsu Scientific and Technical Journal*, vol. 46, no. 1, pp. 47–55, 2010.

[10] L. Cheveau, E. Goray, and R. Salmon, "Watermarking — summary results of EBU tests," http://up.ebu.ch/en/technical/trev/trev_index-eurovision.html, European Broadcasting Union, Tech. Rep., Mar. 2001.

[11] S. Wohlgemuth, I. Echizen, N. Sonehara, and G. Müller, "Tagging disclosures of personal data to third parties to preserve privacy," in *Security and Privacy – Silver Linings in the Cloud*, ser. IFIP Advances in Information and Communication Technology, K. Rannenberg, V. Varadharajan, and C. Weber, Eds. Springer Boston, 2010, vol. 330, pp. 241–252, 10.1007/978-3-642-15257-3_22.

[12] R. Accorsi, "Bbox: A distributed secure log architecture," in *European Workshop on Public Key Services, Applications and Infrastructures, to appear ser. Lecture Notes in Computer Science. Springer*, 2011.

[13] A. Pretschner, M. Hilty, F. Schutz, C. Schaefer, and T. Walter, "Usage control enforcement: Present and future," *Security & Privacy, IEEE*, vol. 6, no. 4, pp. 44–53, 2008.

[14] J. Rutkowska, "Subverting VistaTM kernel for fun and profit," *SyScan and Black Hat (Briefings)*, 2006.

[15] R. Naraine, "Blue pill prototype creates 100% undetectable malware," http://www.eweek.com/c/a/Windows/Blue-Pill-Prototype-Creates-100-Undetectable-Malware/, Jun. 2006.

[16] R. Meersman, Z. Tari, W. Mazurczyk, and K. Szczypiorski, "Steganography of VoIP streams," in *On the Move to Meaningful Internet Systems: OTM 2008*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2008, vol. 5332, pp. 1001–1018.

[17] E. Wustrow, S. Wolchok, I. Goldberg, and J. A. Halderman, "Telex: Anticensorship in the network infrastructure," in *Usenix Security Symposium*, 2011.

[18] J. Barbier, É. Filiol, and K. Mayoura, "Universal jpeg steganalysis in the compressed frequency domain," in *Digital Watermarking*, ser. Lecture Notes in Computer Science, Y. Shi and B. Jeon, Eds. Springer Berlin / Heidelberg, 2006, vol. 4283, pp. 253–267, 10.1007/11922841_21.

[19] T. Pevný and J. Fridrich, "Towards multi-class blind steganalyzer for JPEG images," in *Digital Watermarking*, M. Barni, I. Cox, T. Kalker, and H. Kim, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, vol. 3710, pp. 39–53.

[20] A. Church, "A Note on the Entscheidungsproblem," *The Journal of Symbolic Logic*, vol. 1, pp. 40–41, 1936.

[21] H. G. Rice, "Classes of recursively enumerable sets and their decision problems," *Transactions of the American Mathematical Society*, vol. 74, no. 2, pp. pp. 358–366, 1953.

[22] B. Borchert and F. Stephan, "Looking for an analogue of rice's theorem in circuit complexity theory," *Mathematical Logic Quarterly*, vol. 46, no. 4, pp. 489–504, 2000.

[23] N. Hopper, L. von Ahn, and J. Langford, "Provably secure steganography," *Computers, IEEE Transactions on*, vol. 58, no. 5, pp. 662 –676, may 2009.

[24] J. Barda and L. Cheveau, "Access control and watermarking," http://up.ebu.ch/en/technical/trev/trev_index-eurovision.html, European Broadcasting Union, Tech. Rep. 282, Mar. 2000.

[25] Y. L. Simmhan, B. Plale, and D. Gannon, "A survey of data provenance in e-science," *ACM SIGMOD Record*, vol. 34, no. 3, p. 31, Sep. 2005.