VOL. 11, NO. 15, AUGUST 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

PREVENTION OF SURREPTITIOUS DENIAL OF SERVICE IN CLOUD COMPUTING

K. Sathiyapriya and Uday Bhaskar Reddy
Computer Science and Engineering, SRM University, Chennai, Tamilnadu, India
E-Mail: Priya.kanaiyan@gmail.com

ABSTRACT

An approach to orchestrate stealthy assault patterns, which exhibit a slowly-increasing-depth pattern, designed to inflict the maximum financial rate to the cloud consumer, while respecting the job measurement and the carrier arrival cost imposed by way of the detection mechanisms. We describe both find out how to apply the proposed technique, and its results on the target process deployed within the cloud. In precise, we suggest an object-centered approach that enables enclosing our logging mechanism at the side of users' information and policies. We leverage the JAR programmable capabilities to both create a dynamic and touring object, and to make certain that any entry to users' knowledge will set off the authentication and the automated logging to the JARs. To give a boost to consumer's manage, we also furnish allotted auditing mechanisms. We furnish broad experimental studies that show the efficiency and effectiveness of the proposed systems.

Keywords: cloud computing, accountability, data sharing.

1. INTRODUCTION

The proposed assault method, particularly SIPDAS (Slowly growing-Polymorphic DDoS attack strategy) may also be utilized to a number of style of attacks, that leverage identified application vulnerabilities, with the intention to degrade the carrier supplied by way of the goal utility server going for walks within the cloud. The term polymorphic is prompted to polymorphic assaults which exchange message sequence at every successive illness with the intention to steer clear of signature detection mechanisms. Although the sufferer detects the SIPDAS assault, the assault strategy will also be re-provoke via using one more application vulnerability (polymorphism in the type), or an extra timing (polymorphism over time). To validate the stealthy characteristics of the proposed SIPDAS assault, we explore talents options proposed in the literature to detect sophisticated low-price DDoS assaults. We show that the proposed slowly-increasing polymorphic behavior induces sufficient overload on the target system (to motive a significant financial losses), and evades, or nonetheless, delays extensively the detection methods. Moreover, in order to explore the assault impact in opposition to a utility deployed in a cloud environment, this paper makes a speciality of probably the most critical threats to Cloud Computing, which comes from XML situated DoS (X-DoS) attacks to the net-founded systems. The experimental test is based on mainly mosaic framework, which offers both a 'software Platform', that allows the execution of purposes developed making use of the mosaic API, and a 'Cloud agency', that acts as a provisioning method, brokering assets from a federation of cloud vendors. We advise a novel computerized and enforceable logging mechanism within the cloud. To our talents, this is the first time a systematic procedure to knowledge accountability by means of the novel utilization of JAR files is proposed. The main proposed architecture is platform independent and totally decentralized, so not required the dedicated authentication or storage method in place. We go beyond common entry manage in that we provide a special degree of usage control for the blanketed information after these are dropped at the receiver. We habits experiments on an actual cloud testbed. The results demonstrate the affectivity, scalability, and granularity of our strategy. We additionally provide a exact security evaluation and discuss the reliability and force of our architecture. This paper is an extension of our earlier conference paper [40]. We now have made the following new contributions. First, webuilt-in integrity checks and oblivious hashing (OH) method to our approach with the intention to beef up the dependability of our system in case of compromised JRE. We additionally up-to-date the log documents constitution to provide further ensures of integrity and authenticity. 2d, we accelerated the protection evaluation to cover extra viable attack situations. 1/3, we report the results of new experiments and provide a radical evaluation of the system performance. Fourth, we've got added a certain dialogue on associated works to arrange readers with a greater working out of historical past potential. Sooner or later, now we have improved the presentation by means of including more examples and illustration graphs.

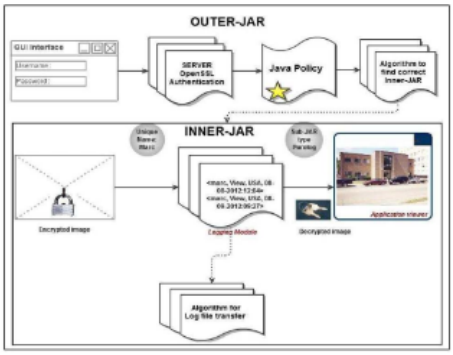
2. SLOWLY-INCREASING-POLYMORPHIC DDOS ATTACK PATTERN AGAINST CLOUD APPLICATION

The approach implemented by each and every agent to perform stealthy service degradation in the Cloud Computing. It has been specialized for an X-DoS attack. Specifically, the attack is performed by injecting polymorphic bursts of length T with the value of increasing intensity until the main attack is either successful or detected. Each burst is formatted in such a way as to inflict a certain average level of load C. In particular, fixed the maximum number of nested tags (tag Threshold), the routine pick Random Tags (...) randomly returns the number of nested tags n for each message (row 4). Based on n, the routine compute Interarrival Time uses

a specific algorithm to compute the inter-arrival time for injecting the message which is next (row 5). Then at the end of the period T, if the condition 'attack Successful' is false (evaluated by the Meter which describes in the next Sec. V-A2), attack intensity is increased (row 10). If the condition shows the message as 'attack Successful' is true, the attack intensity is maintained constant until either the attack is detected, or the auto-scaling mechanism enabled in the cloud adds new cloud resources. The attack is performed until the attack is either detected, or the main average message rate of the next burst to be injected is greater than the d In this last case, the Agent notifies to the Master that the maximum average message rate is reached (row 26), and it continues to inject the messages formatted according to the last level of load C reached (row 30). In this section, we mainly analyzes the effect of an X-DoS attack which occurs on the cloud system in terms of provided quality of service. The mosaic-based test bed designed to test the attack effectiveness consists of two independent cloud applications, the target application server namely SUA and the 'Attacker'. Both applications are mainly designed according to a mosaic paradigm and runs on the independent set of VMs. The SUA application emulates a typical XML-based on Web application which receives the XML messages via HTTP, and performs the XML parsing (using an approach called as DOM), and some other simple elaborations (e.g., it computes all the total number of nested tags). The mosaic implementation of such a scenario is represented in A HTTP Cloudlet manages the HTTP messages and forwards them to the Analyzer which is considered as XML Analyzer, which parses the XML document, and it stores the results (in a Key-Value store). The advantage of using mosaic is, we can able to scale the application automatically when the virtual node is overloaded (starting new VMs). Moreover, the best is about the mosaic monitoring tools [11], because we are able to evaluate the resource consumption of each and every VM which is involved and the number of retrieved messages (XML documents) to be processed. In [31], we adopted the TPC Benchmark W (TPC-W) is mainly adopted to assess the effectiveness of the X-DoS attack with respect to the provided quality of service [33]. It is a transactional Web benchmark is used to simulate activities of the business oriented transactional Web server (i.e., the multiple transaction types execution that span a breadth of the complexity). The each Web interaction shows a different Computational cost, and it subjects to the response time constraint. The performance metric which reported by TPC-W is the number of Web interactions processed per second, which are named as WIPS. In this overview, target application which exhibits is an ad-hoc Web service, so we cannot use the TPC-W workload. In order to perform these similar existing evaluations, we have built a TPCW emulator, which generates the same workload for the SUA application. Moreover, it calculates and measures the number of operations processed per second (WIPS), then re-transmits same request to the application when it is not processed before a fixed timeout. It should be clear that this testbed does not aim at the correctly emulating TPC-W benchmark. It only uses for the basis to build up a general workload, under which we are easily able to analyze the attack effect.

3. AUTOMATED LOGGING MECHANISM

We leverage the computerized capability of JARs to participate in computerized signing. A logger detail is a stacked espresso JAR understanding file which stores a consumer's knowledge items and corresponding log understanding documents. As demonstrated in Fig, our advised JAR understanding file involves one external JAR attaching one or more internal JARs.



The structure of the JAR file

The predominant authorized accountability of the external JAR is to control verification of businesses which desire to accessibility the knowledge saved within the JAR data file. In our standpoint, the understanding entrepreneurs would probably not appreciate the exact CSPs which can be going to handle the expertise. Consequently, verification is designated in maintaining with the servers' performance (which we suppose to be well-known through a search provider), however than the server's URL or identification. For example, a plan may state that Server X is authorised to acquire the know-how if it's a storage space server. As recounted beneath, the outside JAR may additionally have the accessibility administration efficiency to put in force the expertise proprietor's necessities, specific as espresso strategies, on making use of the know-how. A espresso plan identifies which authorizations are available for a unique piece of rule in a espresso program atmosphere. The authorizations indicated in the espresso plan are in phrases of File program Permissions. Nonetheless, the expertise proprietor can specify the authorizations in character-centric occasions not like the natural code-centric protection supplied via coffee, utilising coffee Authentication and Permission options. Moreover, the outside JAR can also be in cost of picking out the right inside JAR in maintaining with the identification of the brand who needs the expertise. Every inside JAR comprises the secured capabilities, class data files to accomplish treatment of log abilities files and show off surrounded talents in a suitable structure, and a log data declare every secured product. We help two options:

PureLog: Its foremost system is to historical past each accessibility the know-how. The log knowledge documents are used for exact audit purpose.

AccessLog: It has two functions: signing movements and implementing accessibility administration. In main issue an accessibility demand is declined, the JAR will historical earlier the time when the demand is made. If the accessibility demand is furnished, the JAR will additionally historic past the accessibility expertise along with the scale for which the accessibility is authorized. The two types of signing segments permit the knowledge proprietor to implement distinctive accessibility situations each proactively (in concern of entry Logs) or reactively (in concern of Pure Logs). For instance, choices like repayments could have got to use Pure working out. Access figuring out maybe principal for choices which have got to put into outcomes carrier-stage contracts equivalent to restricting the publicity to a couple delicate content material at a given region. To bring out these elements, the within JAR includes a category information declare composing the log files, but a different class know-how file which goes with the log harmonizer, the secured figuring out, a third category understanding declare displaying or striking in the talents (founded on whether or not now we've got a PureLog, or an Access Log), and most of the people key of the IBE key couple that is quintessential for encrypting the log records. No key main explanations are ever saved in it. The outside JAR may incorporate a number of internal JARs, moreover to a class information declare authenticating the online servers or the shoppers, an extra class knowledge file discovering the proper internal JAR, a third category data file which assessments the JVM's credibility using unaware hashing. Extra, a category information file is used for dealing with the GUI for customer verification and the espresso insurance policy.

3.1 Dependability of logs

3.1.1 JARs availability

To avert strikes perpetrated on off-line JARs, the CIA has a log harmonizer which has two most important duties: to handle duplicates of JARs and to revive damaged files. Every log harmonizer is in manipulate of duplicates of logger factors containing the identical set of major aspects merchandise. The harmonizer is utilized as a JAR knowledge file. It does not comprise the purchaser's small print merchandise being audited, however includes type data files for both a server and a purchaser procedures to allow it to glue with its logger elements. The harmonizer retailers mistake amendment small print sent from its logger factors, as good because the purchaser's IBE decryption key, to decrypt the log important points and control any copy important points. Replica important points influence from duplicates of the person's details JARs. On account that consumer's important elements are particularly along with the logger facet in a small print JAR expertise file, the logger can be duplicated along side the individual's small print. Thus, the manufacturer new duplicate the logger includes the historic log small print headquartered on the usage of important points within the targeted details JAR information file. Such ancient log details are repetitive and unrelated to the manufacturer new duplicate of the details. To reward the main points proprietor a view, the harmonizer will mix log small print from all duplicates of the principal points JARs with the help of disposing of redundancy. For getting better explanations, logger factors are required to supply mistake change most important facets to the harmozer after writing each and every log historical past. Thus, logger factors continuously known as ping the harmonizer earlier than they enable any accessibility right. If the harmonizer isn't to be had, the logger elements will refuse all accessibility. On this approach, the harmonizer inhibits strikes which try and hold the predominant aspects JARs off-line for unseen utilization. If the enemy took the details JAR off-line after the harmonizer was once once pinged, the harmonizer nonetheless has the error modification major features about this accessibility and may quickly to find the dropping history. In case of crime of JAR documents, the harmonizers will restoration the documents with the support of Reed-Solomon mistake amendment rule [45]. Almost always, each person signing JAR, when created, involves a Reed-Solomon-based encoder. For every n signs within the log talents file, n redundancy signs are delivered to the log harmonizer within the style of pieces. This creates an mistake fixing rule of size 2n and allows for the mistake amendment to determine and proper n errors. We choose the Reed Solomon rule as it accomplishes the equal rights in the Singleton limited [36], making it a excellent viable type separable rule and as a final result results in an absolute first-rate possible mistake modification. The log harmonizer is founded at a identified IP maintain. Most often, the harmonizer exists on the character's finish as a part of his nearby pc, or then again, it would each be saved in consumer's computer or in a proxies server.

4. END-TO-END AUDITING MECHANISM

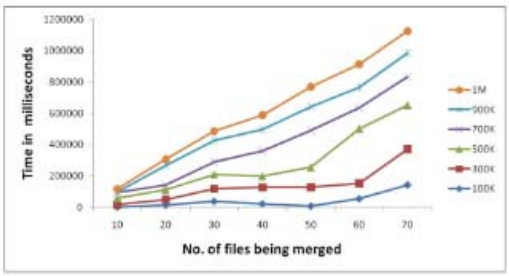
4.1 Push and pull mode

To enable customers to be suitable and flawlessly steered about their know-how utilization, our distributed signing procedure is accompanied by means of a modern-day audit approach. We support two helping audit modes: 1) power mode; 2) take method. Push procedure. In this approach, the information are mainly compelled to the understanding proprietor (or auditor) by means of the harmonizer. The drive motion is often activated by way of both form of the subsequent two activities: one is that the three hundred and sixty five days progresses for a special period regular with the transitority clock placed as part of the JAR file; the other is that the JAR information file surpasses the dimension exclusive by the content material proprietor at a lot of length of progress. After the understanding are despatched to the skills proprietor, the log know-how records can be thrown out, so that you could free the gap for future accessibility information. In conjunction with the log expertise documents, the error

fixing skills for these understanding can also be thrown out. This drive approach is the fundamental mode which can be carried out through every the PureLog and the AccessLog, despite whether or now not there's a demand from the information proprietor for the log capabilities records. This approach supplies two most important sides within the signing architecture: 1) it guarantees that the dimension the log expertise documents does now not burst and a pair of) it makes it possible for correct realization and modification of any loss or damage to the log talents records. Related to the latter perform, we realize that the auditor, upon receiving the log working out file, will affirm its cryptographic assures, through verifying the files' reliability and credibility. By using development of the understanding, the auditor, might be capable to rapidly verify forgery of figuring out, making use of the checksum offered to each file. Pull procedure. This approach enables auditors to get good the advantage at any time once they need to confirm the latest accessibility their possess potential. The take suggestion comprises fairly quite simply of an FTP take manage, which can be issues from the manipulate line. For innocent purchasers, a informed including a group working out file may also be with no trouble developed. The demand will also be despatched to the harmonizer, and the user possibly advised of the data's areas and acquire a replica of the precise and enclosed log expertise file. Four.2 Algorithms to allow consumers to be correct and absolutely advocated about their important points utilization, our assigned making a option on procedure is related to a trendy assessment method. We aid two helping overview modes: 1) energy mode; 2) take method. Push process. In this approach, the essential elements are on the whole compelled to the main points owner (or auditor) by way of the harmonizer. The vigour movement might be precipitated by either variety of the next two actions: one is that the twelve months advances for a particular interval in preserving with the short-time period time placed as a part of the JAR file; the reverse is that the JAR essential facets computer file exceeds the sizing designated by way of the content material material owner at a kind of length of growth. After the principal aspects are despatched to the main points proprietor, the log small print knowledge will regularly be tossed out, as a way to free the distance for upcoming availability small print. Together with the log important points understanding, the error fixing small print for these essential elements can be tossed out. This energy method is the main mode which can also be utilized with the aid of each the PureLog and the AccessLog, in spite of whether or not there's a requirement from the foremost elements proprietor for the log small print capabilities. This process provides two main facets within the opting for constitution: 1) it ensures that the sizing the log main point understanding does no longer rush and a pair of) it allows for suitable identification and adjustment of any loss or harm to the log small print understanding. Regarding the latter perform, we become aware of that the auditor, after getting the log details laptop file, will validate its cryptographic ensures, via confirming the documents' stability and steadiness. By means of progress of the essential points, the auditormight be in a position to without problems respect forgery of major facets, utilising the checksum integrated to every historical prior. Take method. This process makes it viable for auditors to revive the small print at any time when they have got to verify the ultra-ultra-modern availability their own small print. The take thought entails comfortably of an FTP take administration, which can be issues from the administration line. For easy consumers, a authentic in conjunction with a bunch small print computer file will also be simply designed. The requirement is usually despatched to the harmonizer, and the buyer will normally be recommended of the information's areas and collect a duplicate of the risk-free and surrounded log fundamental elements desktop file.

5. PERFORMANCE STUDY

Inside the assessments, we first analyze time taken to create a log information file and then measure the cost in the procedure. With reference to time, the cost can come up at three features: in the course of the verification, during safeguard of a log record, and for the duration of the consolidating of the files. Additionally, with regard to cupboard space price, we comprehend that our constitution could be very tender and portable, in that the one knowledge to be saved are given by using utilizing the distinctive data documents and the associated documents. Additional, JAR act as an air compressor of the knowledge records that it manages. In distinct, as awarded in, a number of talents files can also be managed by way of using the equal logger aspect. To this degree, we analyze whether or no longer a single logger side, used to handle a couple of data file, effect stored in space for storing fee.

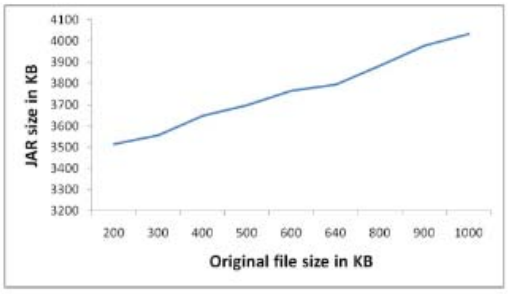


Time to merge log files

Within the first round of checks, we are occupied with discovering out time taken to make a log data file when there are organizations regularly obtaining the data, causing ongoing logging. Outcomes are established in Fig.. It is not shocking to see that plenty of a chance to make a log data file improves linearly with the size of the log information file. Peculiarly, plenty of a hazard to make a a hundred Kb knowledge file is ready 114.5 ms at the same time plenty of a danger to make a 1 MB data file earnings at 731 ms. With this study as the rule, you may come to a decision how lengthy to be distinctive between places, retaining other reasons like area constraints or community traffic in mind. To examine if the log harmonizer is usually a bottleneck, we measure how

lengthy required to combine log knowledge files. In this study, we guaranteed that each and every of the log data files had 10 to 25 percentage of the know-how in average with one different. The specific type of know-how in usual used to be random for each and every repeating of the research. The there was once a time averaged over 10 reps. We proven a lot of a hazard to mix up to 70 log data records of a hundred KB, 300 KB, 500 KB, seven hundred KB, 900 KB, and 1 MB each and every. The results are proven in Figure. We will detect that point improves just about linearly to the style of information records and dimension knowledge records, with the least moment taken for consolidating two one hundred KB log knowledge files at fifty nine ms, even as plenty of a risk to combine 70 1

MB knowledge files was once 2.35 minutes.



Finally, we evaluate whether or no longer only one logger, used to control a couple of working out file, effect in storage price. We measure the dimension the loggers (JARs) with the support of particular the number and dimension know-how merchandise organised via them. We founded the develop in dimension the logger containing 10 fabric knowledge documents (i.e., graphics) of the equal dimension on the grounds that the pleasant improves. Naturally, in case of larger dimension expertise products organised with the aid of a logger, the whole logger moreover improves in dimension. The dimension logger grows from three, 500 to four, 035 KB when the dimension fabric merchandise changes from 200 KB to 1 MB. Overall, as a result of the pressure supplied by way of JAR understanding documents, the dimension the logger is dependent upon the dimension the primary potential files it entails. Detect that we deliberately failed to incorporate tremendous log potential documents (lower than 5 KB), to be equipped to middle of concentration on the rate introduced by way of having multiple material capabilities records in just one JAR.

6. CONCLUSIONS

We told impressive approaches for immediately signing any entry to the talents within the reasoning along side an audit process. Our process permits the abilities proprietor to no longer handiest overview his fabric however in addition put in force strong back-finish protection if needed. In addition, traditionally essentially the most major sides of our work is that it makes it possible for the information proprietor to investigate even these duplicates of its advantage that have been made without any knowledge. At the same time, we approach to enhance our method to verify the reliability of the JRE and the authentication of JARs [23]. For illustration, wewill evaluation whether or not it is possible to utilize the idea of a blanketed JVM [18] being designed via IBM. This research is specific at supplying program mess stage of skills to deal with espresso programs. In the end, we manner to design a tremendous and further typical object-oriented approach to obtain unbiased security of journeying fabric. We want to back up a massive kind of protection guidelines, like guidelines directions for written textual content material information records, utilization manipulate for executables and general accountability and provenance manages.

REFERENCES

[1] P. Ammann and S. Jajodia. 1993. Distributed Timestamp Generation inPlanar Lattice Networks. ACM Trans. Computer Systems. 11: 205-225.

[2] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson and D. Song. 2007. Provable Data Possession at Untrusted Stores. Proc. ACM Conf. Computer and Comm. Security. pp. 598609.

[3] E. Barka and A. Lakas. 2008. Integrating Usage Control with SIP-Based Communications. J. Computer Systems, Networks and Comm. 2008: 1-8.

[4] D. Boneh and M.K. Franklin. 2001. Identity-Based Encryption from the Weil Pairing. Proc. Int'l Cryptology Conf. Advances in Cryptology. pp. 213-229.

[5] R. Bose and J. Frew. 2005. Lineage Retrieval for Scientific Data Processing: A Survey. ACM Computing Surveys. 37: 128.

[6] P. Buneman, A. Chapman and J. Cheney. 2006. Provenance Managementin Curated Databases. Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '06). pp. 539-550.

[7] B. Chun and A.C. Bavier. 2004. Decentralized Trust Management and Accountability in Federated Systems. Proc. Ann. Hawaii Int'l Conf. System Sciences (HICSS).

[8] OASIS Security Services Technical Committee. 2012. Security AssertionMarkup Language (saml) 2.0. http://www.oasisopen.org/committees/tchome.php?wg abbrev=security.

[9] R. Corin, S. Etalle, J.I. den Hartog, G. Lenzini and I. Staicu. 2005. ALogic for Auditing Accountability in Decentralized Systems. Proc. IFIP TC1 WG1.7 Workshop Formal Aspects in Security and Trust. pp. 187-201.

[10] B. Crispo and G. Ruffo. 2001. Reasoning about Accountability within Delegation. Proc. Third Int'l Conf. Information and Comm. Security (ICICS). pp. 251-260.

[11] Y. Chen et al. 2003. Oblivious Hashing: A Stealthy Software Integrity Verification Primitive. Proc. Int'l Workshop Information Hiding, F. Petitcolas, ed. pp. 400-414.

[12] S. Etalle and W.H. Winsborough. 2007. A Posteriori Compliance Control. 2007. SACMAT '07: Proc. 12th ACM Symp. Access Control Models and Technologies. pp. 11-20.

[13] X. Feng, Z. Ni, Z. Shao and Y. Guo. 2007. An Open Framework for Foundational Proof-Carrying Code. Proc. ACM SIGPLAN Int'l Workshop Types in Languages Design and Implementation. pp. 67-78.

[14] Flickr, http://www.flickr.com/, 2012.

[15] R. Hasan, R. Sion and M. Winslett. 2009. The Case of the Fake Picasso: Preventing History Forgery with Secure Provenance. Proc. Seventh Conf. File and Storage Technologies. pp. 1-14.

[16] J. Hightower and G. Borriello. 2001. Location Systems for Ubiquitous Computing. Computer. 34(8): 57-66.

[17] J.W. Holford, W.J. Caelli, and A.W. Rhodes. 2004. Using Self Defending. Objects to Develop Security Aware Applications in Java. Proc. 27th Australasian Conf. Computer Science. 26: 341-349.

[18] Trusted Java Virtual Machine IBM, http://www.almaden.ibm.com/cs/projects/jvm/, 2012.

[19] P.T. Jaeger, J. Lin and J.M. Grimes. 2009. Cloud Computing and Information Policy: Computing in a Policy Cloud? J. Information Technology and Politics. 5(3): 269-283.

[20] R. Jagadeesan, A. Jeffrey, C. Pitcher and J. Riely. 2009. Towards a Theory of Accountability and Audit. Proc. 14th European Conf. Research in Computer Security (ESORICS). pp. 152-167.

[21] R. Kailar. 1996. Accountability in Electronic Commerce Protocols. IEEE Trans. Software Eng. 22(5): 313-328.

[22] W. Lee, A. CinziaSquicciarini and E. Bertino. 2009. The Design and Evaluation of Accountable Grid Computing System. Proc. 29th IEEE Int'l Conf. Distributed Computing Systems (ICDCS '09). pp. 145-154.

[23] J.H. Lin, R.L. Geiger, R.R. Smith, A.W. Chan and S. Wanchoo. 2004. Method for Authenticating a Java Archive (jar) for Portable Devices, US Patent 6,766,353.

[24] F. Martinelli and P. Mori. 2010. On Usage Control for Grid Systems. Future Generation Computer Systems. 26(7): 1032-1042.

[25] T. Mather, S. Kumaraswamy and S. Latif. 2009. Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance (Theory in Practice), first ed. O' Reilly.

[26] M.C. Mont, S. Pearson, and P. Bramhall. 2003. Towards Accountable Management of Identity and Privacy: Sticky Policies and Enforceable Tracing Services. 2003. Proc. Int'l Workshop Database and Expert Systems Applications (DEXA). pp. 377-382.

[27] S. Oaks, Java Security. O'Really, 2001.

[28] J. Park and R. Sandhu. 2002. Towards Usage Control Models: Beyond Traditional Access Control. SACMAT '02: Proc. Seventh ACMSymp. Access Control Models and Technologies. pp. 57-64.

[29] J. Park and R. Sandhu. 2004. The Uconabc Usage Control Model. ACM Trans. Information and System Security. 7(1): 128174.

[30] S. Pearson and A. Charlesworth. 2009. Accountability as a WayForward for Privacy Protection in the Cloud. Proc. First Int'lConf. Cloud Computing.

[31] S. Pearson, Y. Shen, and M. Mowbray. 2009. A Privacy Manager for Cloud Computing. Proc. Int'l Conf. Cloud Computing (CloudCom). pp. 90-106.

[32] A. Pretschner, M. Hilty and D. Basin. 2006. Distributed Usage Control. Comm. ACM. 49(9): 39-44.

[33] A. Pretschner, M. Hilty, F. Schuotz, C. Schaefer, and T. Walter. 2008. Usage Control Enforcement: Present and Future. IEEE Security and Privacy. 6(4): 44-53.

[34] A. Pretschner, F. Schuotz, C. Schaefer, and T. Walter. 2009. Policy Evolution in Distributed Usage Control. Electronic Notes Theoretical Computer Science. 244: 109-123.

[35] NTP: The Network Time Protocol, http://www.ntp.org/, 2012.

[36] S. Roman. 1995. Coding and Information Theory. Springer-Verlag.

[37] B. Schneier. 1993. Applied Cryptography: Protocols, Algorithms, and Source Code in C. John Wiley and Sons.

[38] T.J.E. Schwarz and E.L. Miller. 2006. Store, Forget, and Check: UsingAlgebraic Signatures to Check Remotely Administered Storage. Proc. IEEE Int'l Conf. Distributed Systems. p. 12.

[39] A. Squicciarini, S. Sundareswaran and D. Lin. 2010. Preventing Information Leakage from Indexing in the Cloud. Proc. IEEE Int'l Conf. Cloud Computing.

[40] S. Sundareswaran, A. Squicciarini, D. Lin and S. Huang. 2011. Promoting Distributed Accountability in the Cloud. Proc. IEEE Int'l Conf. Cloud Computing.