NotaLogger: Notarization Code Generator and Logging Service

Maurice HT Ling
School of Chemical and Biomedical Engineering,
Nanyang Technological University, Singapore
Department of Zoology,
The University of Melbourne, Australia
mauriceling@acm.org

Abstract

The act of affixing a signature and date to a document, known as notarization, is often used as evidence for sighting or bearing witness to any documents in question. Notarization and dating are required to render documents admissible in the court of law. However, the weakest link in the process of notarization is the notary; that is, the person dating and affixing his/her signature. A number of legal cases had shown instances of false dating and falsification of signatures. In this study, NotaLogger is proposed, which can be used to generate a notarization code to be appended to the document to be notarized. During notarization code generation, the user can include relevant information to identify the document to be notarized and the date and time of code generation will be logged into the system. Generated and used notarization code can be verified by searching in NotaLogger, and such search will result in date time stamping by a Network Time Protocol server. As a result, NotaLogger can be used as an "independent witness" to any notarizations. NotaLogger can be accessed at http://mauricelab.pythonanywhere.com/notalogger/.

1. Introduction

The act of affixing a signature and date to a document is commonly known as notarization, which is often used as evidence for sighting or bearing witness to any documents in question (Crystal and Giannoni-Crystal, 2012). Notarization and dating are important aspects to render documents admissible in the court of law as suggested in All Points Capital Corporation v. Boyd Brothers Incorporated (2011). However, the weakest link in the process of notarization is the notary; that is, the person dating and affixing his/her signature. There had been a number of legal cases pertaining to signatures and dating.

There had been legal cases pertaining to wrong or false dating, such as backdating a document. For example, Alvarez v. Target Corporation (2007) stated, "false dating by a notary employee of the trustee in a nonjudicial foreclosure is an unfair or deceptive act or practice and satisfies the first three elements under the Washington CPA". People v. Susalla (1974) cited Perkins (1969) that "forgery may be committed, for example, by one using his or her own name, by false dating, or using one's name as that of another". Mortgage Capital Resource Corporation false dated documents pertaining to loan applications (Bryant v. Mortgage Capital Resource Corporation, 2002).

Similarly, the authenticity of signatures can also be questioned. There had been many instances of falsifying signatures. For example, Richard and Tania Eicoff misappropriated

funds from the estate of John Rouson by forging signatures on checks (Rouson v. Eicoff, 2006). Klem v. Washington Mutual Bank (2013) states that the "court does not take lightly the importance of a notary's obligation to verify the signor's identity and the date of signing by having the signature performed in the notary's presence" and that "the act of false dating by a notary employee of the trustee in a nonjudicial foreclosure is an unfair or deceptive act or practice and satisfies the first three elements under the Washington CPA." Historically, a method to authenticate signatures had been to affix a symbol of personal artefact, such as a personal seal (Spear, 2005) or any inscribed objects (White and Beaudry, 2009), onto the signature.

Such forgeries had led to disciplinary actions against several attorneys. For example, Jerome J. Holmay, an Attorney at Law of the State of Minnesota, USA, had been suspended for 30 days for forgery of signatures (Matter of Discipline of Holmay, 1987). Joseph Kaminsky, an Attorney at Law of the State of Minnesota, USA, had been suspended for 30 days for forgery of signatures on three separate affidavits and made arrangements for those forged signatures to be notarized (Matter of Discipline of Kaminsky, 1987). Robert H. Aitken, III, an Attorney at Law of the State of Minnesota, USA, had been suspended for 90 days for forging a signature on the plea petition and filing it with the district court (Petition for Disciplinary Action against Aitken, 2013).

In this study, a notarization code generator and logging service, NotaLogger, is proposed. Similar notarization technologies had been proposed to reduce legal risks in e-commerce and contractual activities (Wang, 2011). NotaLogger can be used to generate a random string, known as a notarization code, which can be appended to the document to be notarized. During notarization code generation, the user can include relevant information to identify the document to be notarized and the date and time of code generation will be logged into the system. Generated and used notarization code can be verified by searching in NotaLogger, and such search will result in date time stamping by a Network Time Protocol server, which acts as another layer of verification. As a result, NotaLogger can be used as an "independent witness" to any notarizations. NotaLogger can be accessed at http://mauricelab.pythonanywhere.com/notalogger/.

2. Using NotaLogger

NotaLogger is built on web2py (Di Pierro, 2009), a Python web framework, as an application plug-in and adhering to the model-view-controller (MVC) paradigm. Web2Py had been used in secured applications, such as CyNote (Ng and Ling, 2010), as it contains security features that prevent database injections[1] (Di Pierro, 2009). This is crucial to prevent two different forgeries. Firstly, it prevents potential users from injecting a notarization code that was not previously generated by NotaLogger into the database. This may happen when users want to backdate a notarization. In addition to injection prevention, each activity to NotaLogger's database is assigned an auto-incremental ID. Hence, unless the user as superuser access to the server, such injection and subsequent trail coverage is unlikely. Secondly, it prevents changing details pertaining to a notarization code. This may happen

[1] http://www.pythonsecurity.org/wiki/web2py/

when users want to reuse notarization codes; thereby, reassigning previously generated notarization codes by changing the details supplied when the notarization code was previously generated. This can only happen when the user has either superuser access to the server or administrator access to the web2py installation. As generic details can be provided for notarization code generation, this also suggests that sufficient details pertaining to the use of each generated notarization code should be provided to accurately identify the purpose of each notarization code generation.

The main function of NotaLogger is to generate a notarization code (Figure 1) and log each code generation, together with user supplied details, into a SQLite database. The current version of NotaLogger does not require users to log in before use. It will be the responsibility of the user to provide accurate and sufficient details, excluding any confidential or sensitive information, to identify the purpose of notarization code generation as provided details will be listed as search results. Insufficient details may result in future invalidation of the notarization code if it cannot be uniquely identified. For example, if the user stated that the use of the generated code is to "*notarize letter from Mr. A to Mr. B, HR Manager of Company X, dated 14 October 2013; Reference 2013/HR/00785*", which may provide sufficient identification details than a purpose stating "*notarize letter #1143*", unless there is proper and unequivocal record of Letter #1143. Notarization codes can be of different lengths with a lower limit of 5 characters. Only specific lower case alphabets are used in notarization codes to prevent confusion between lower and upper case alphabets and between "O" and "0" (zero). Forty-nine characters and symbols are used: `'1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'a', 'b', 'd', 'e', 'g', 'h', 'q', 'r', '=', '#', '$', '%', '&'`, and `'@'`.

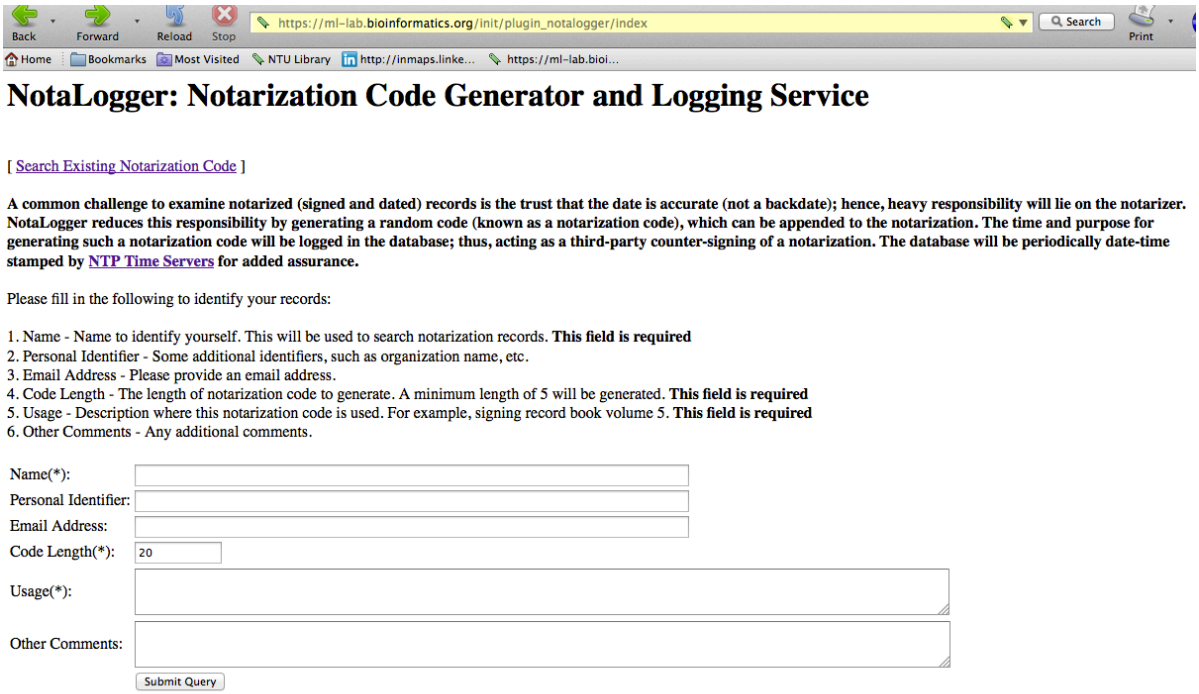

Figure 1. Landing (Initial) Page to Generate a Notarization Code.

As an example, a 20-character notarization code, `55R8UX$Q4@&b8DWR#6dQ`, is generated for the purpose of "*Example for NotaLogger*" (Figure 2 and 3). The system date and time at which the notarization code is generated is given as number of seconds since Unix epoch, which is defined as the number of seconds elapsed since 1st January 1970 midnight GMT. Assuming sufficient identification details are provided; the notarization code - `55R8UX$Q4@&b8DWR#6dQ` – can be appended or written onto the corresponding document.

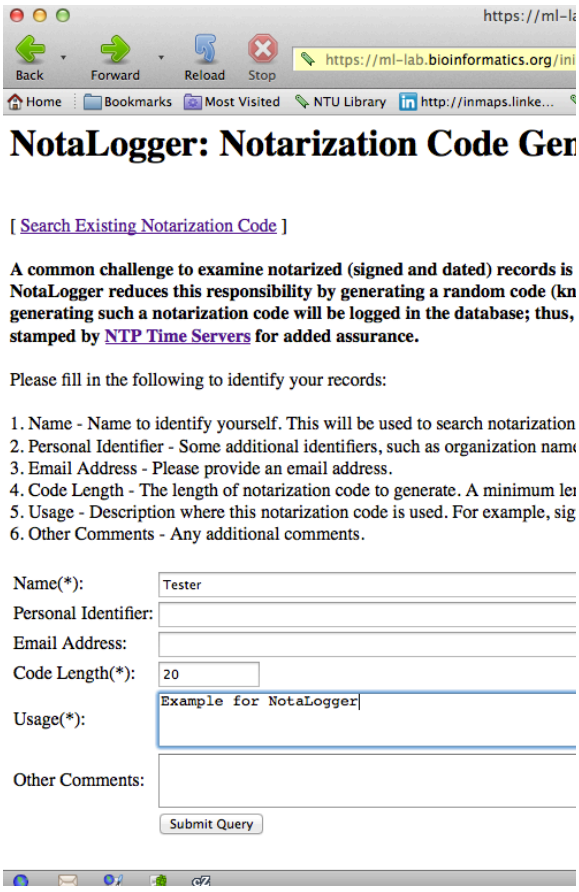

Figure 2. Example Generation of a Notarization Code.

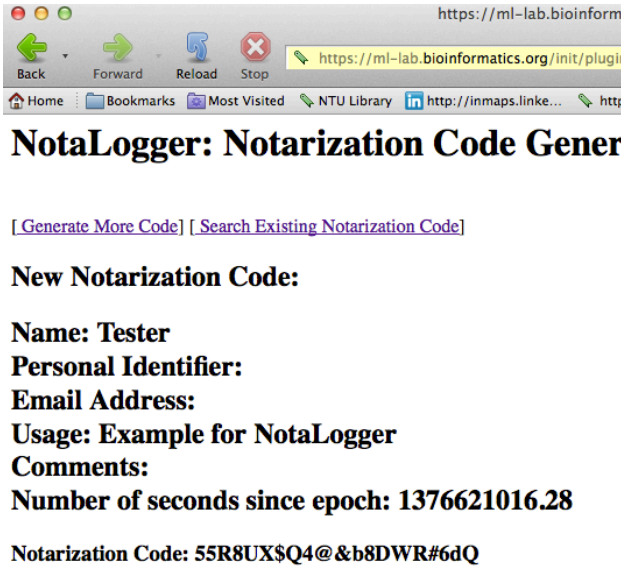

Figure 3. Notarization Code Generated.

A search function is provided for users to search and verify a notarization code (Figure 4). Basically, it retrieves details initially provided at notarization code generation using the entire or part of a notarization code as search term. More than one notarization code may be retrieved using a short search term. In this case, only the first 50 notarization codes and its associated details will be displayed. As the details associated with a notarization code is crucial in identifying the document or the purpose of original notarization code generation and at the same time, displayed as search results (Figure 5); it is crucial that no confidential information be associated with a notarization code.

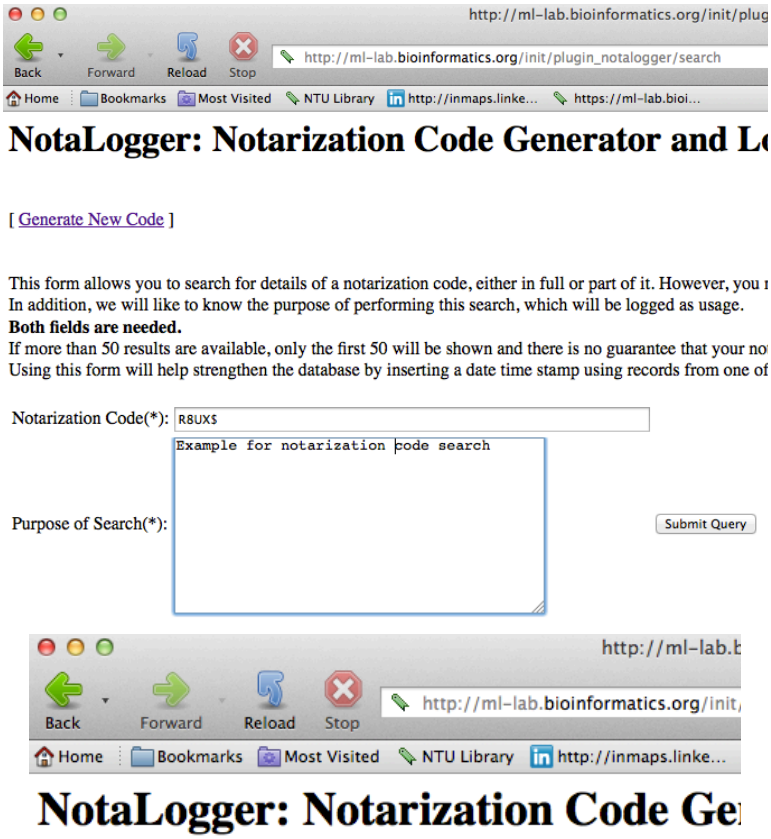

Figure 4. Notarization Code Search Page. The purpose of the current search will be logged together with the search results (see Figure 9).

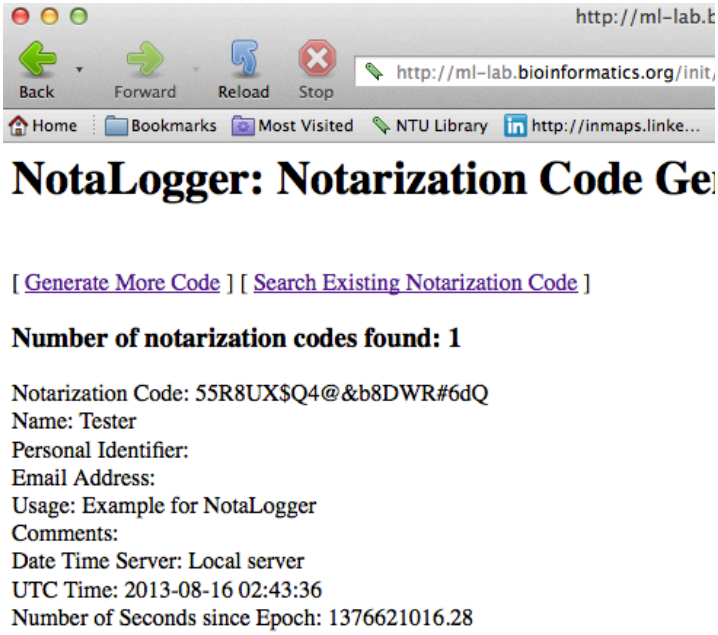

Figure 5. Notarization Code Search Results.

Given a generated notarization code (Figure 6 and 7), each use of the search function will result in date-time stamping of the database using NTP server pool (Figure 8) unless there is a network or connection error. This can act as both date-time stamping of the database by an external party, as well as calibrating the system time (which is used to date-time stamp each notarization code generation) to an NTP server. The connection to an NTP server pool is made using Python NTP library version 0.3.1 (Natali, 2013), which provides information on the IP of the specific NTP server within the server pool, receiving and transmission time, and delay in the time server (Figure 8).

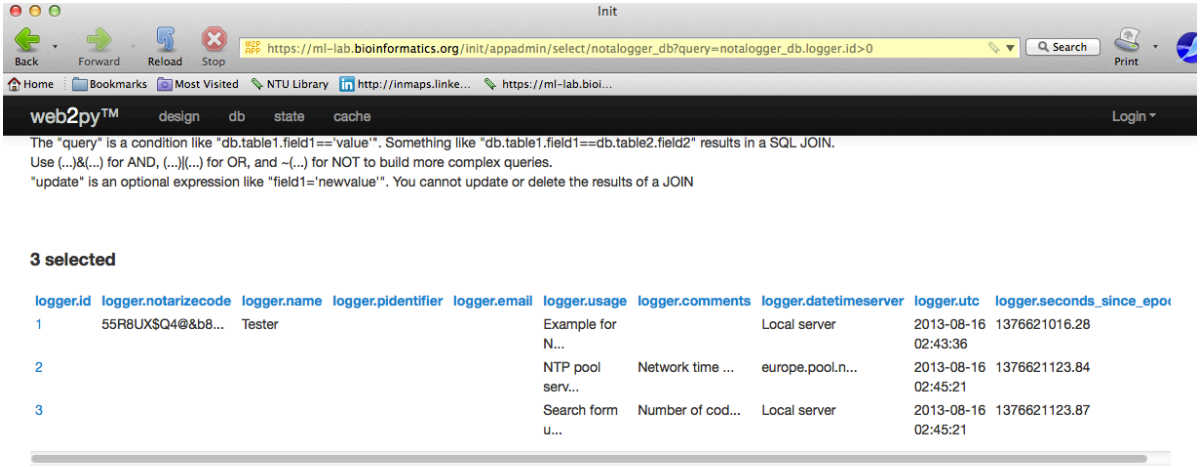

Figure 6. Database after One Cycle of Notarization Code Generation and Search.

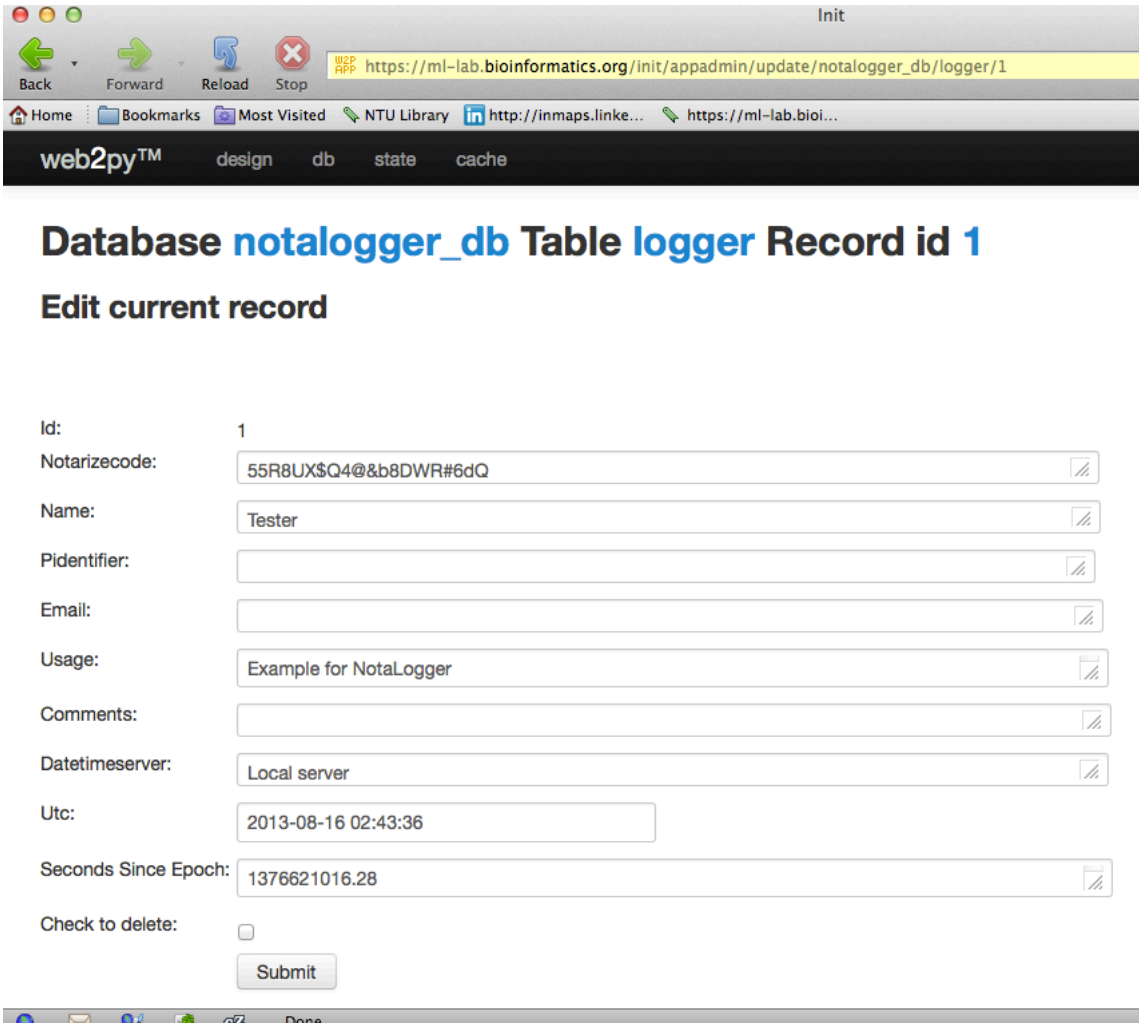

Figure 7. Detailed View of Logging for Notarization Code Generation.

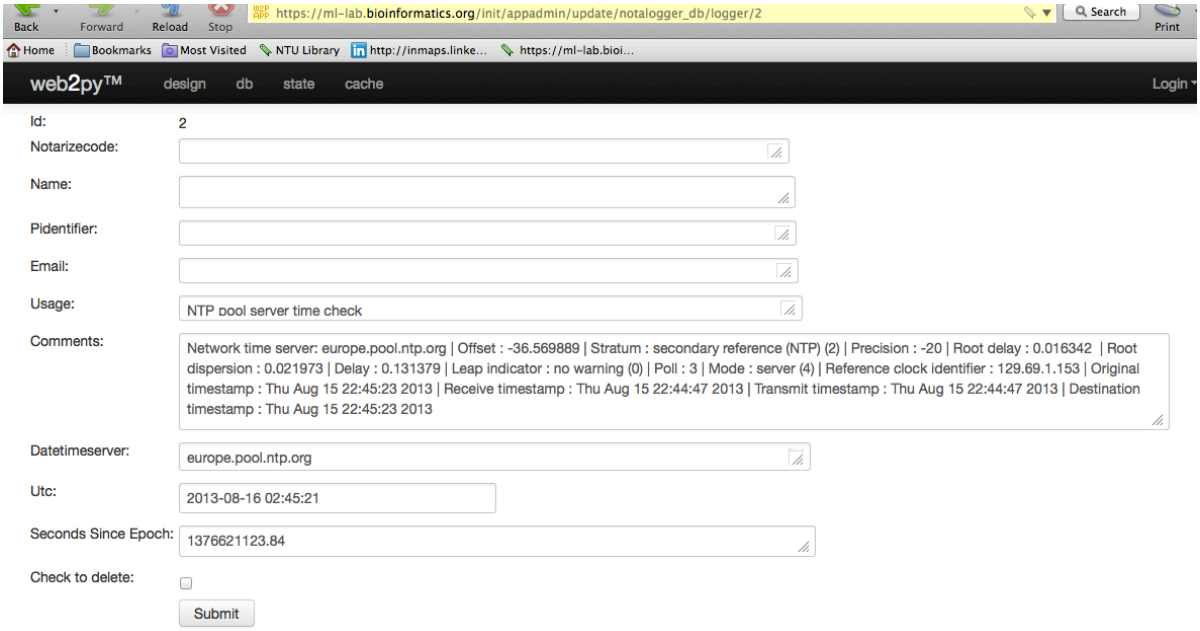

Figure 8. Detailed View of NTP Datetime Stamping.

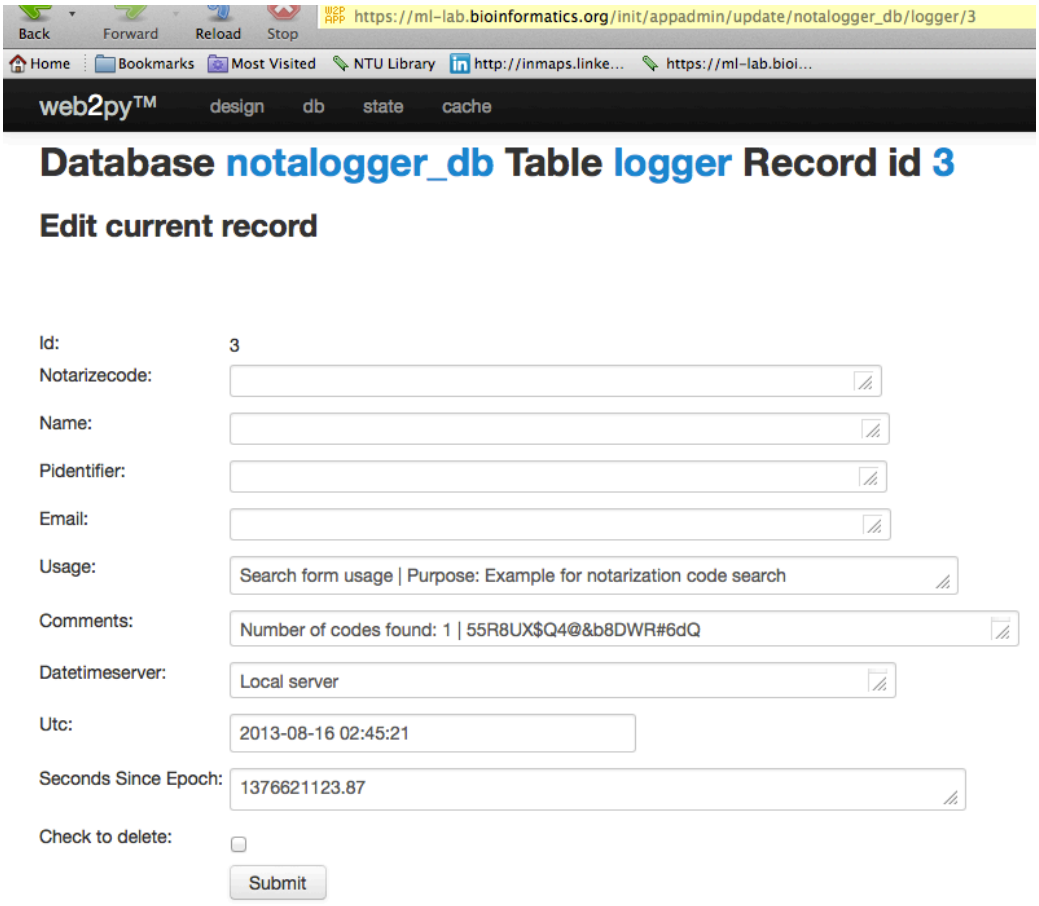

Figure 9. Detailed View of Search Logging. Date and time of the search, as well as the purpose of the search (see Figure 4) and the list of notarization codes found will be logged.

The search and the subsequent results, including the list of notarization codes found, will be logged in the database (Figure 9). Hence, each search record can act as a complete existential validation of one or more notarization codes whenever less than 50 notarization codes were found by to have a specific string provided by the search. That is, listed notarization codes can be deemed to exist and notarization codes that are not listed in the search are non-existent. This is not the case when the logged search result has 50 notarization codes. If the logged search result has 50 notarization codes, this will be an incomplete existential validation of existing notarization codes as only listed notarization codes can be deemed to exist but this does not suggest that notarization codes that are not listed in the search are non-existent.

As more searches are carried out, the abovementioned existential will be approach a cumulative effect as proposed by Lekkas and Gritzalis (2004). That is, different searches over time can repeatedly validate the presence of a notarization code. Hence, the act of searching for a notarization code has an impact on checking the existence of one or more notarization codes, which can act as another layer of fraud prevention to the entire system.

3. Conclusion

The act of notarization by handwritten signature is crucial for sighting or bearing witness to any documents in question (Crystal and Giannoni-Crystal, 2012) but largely dependent on the trustworthiness of the notary (Lekkas and Gritzalis, 2004). In this study, we proposed a third-party notarization code generator and logger, NotaLogger, as a support tool to facilitate notary acts. NotaLogger generates a code, which can be appended to the notarized document, and user-provided details, which can uniquely identify the purpose and document in question when used appropriately, are recorded and date-time stamped. Subsequently, searching for an existing notarization code will trigger date-time stamping of the entire database, and the search results will be logged in the database as an existential validation of notarization codes. As a result, NotaLogger can be used as an "independent witness" to any notarizations as a trusted third-party, which is similar to a notarization authority for emails proposed by Ekanayake et al. (2003).

References

All Points Capital Corporation v. Boyd Brothers Incorporated. 2011. No. 5: 11-cv-116/RS-EMT. District Court of Florida, United States of America.

Alvarez v. Target Corporation. 2007. No. 13-CV-0150-TOR. Washington District Court, United States of America.

Bryant v. Mortgage Capital Resource Corporation. 2002. 197 F. Supp. 2d 1357. Georgia District Court, United States of America.

Crystal, NM. and Giannoni-Crystal, F. 2012. Do the Right Thing (for your duty of competency): Some Ethical and Practical Thoughts on "Notarization" in International Transactions. Global Jurist, 12(2).

Di Pierro, M. 2009. Web2py: enterprise web framework, 2nd edition. Wiley Publishers.

Ekanayake, H., De Zoysa, K. and Dayarathna, R. 2003. A Notarization Authority for the Next Generation of E-Mail Systems. Proceedings of the 5th International Information Technology Conference, pp. 166 – 170.

Klem v. Washington Mutual Bank. 2013. 295 P.3d 1179, 176 Wash. 2d 771. Supreme Court of Washington, United States of America.

Lekkas, D. and Gritzalis, D. 2004 Cumulative notarization for long-term preservation of digital signatures, Computers & Security 23: 413-424.

Matter of Discipline of Holmay 1987. 399 N.W.2d 564. Supreme Court of Minnesota, United States of America.

Matter of Discipline of Kaminsky. 1987. 407 N.W.2d 670. Supreme Court of Minnesota, United States of America.

Natali, Charles-Francois. 2013. Python NTP library. https://pypi.python.org/pypi/ntplib/0.3.1

Ng, YY and Ling, MHT. 2010. Electronic Laboratory Notebook on Web2Py Framework. In: Peer-Reviewed Articles from PyCon Asia-Pacific 2010. The Python Papers 5(3): 7.

Petition for Disciplinary Action against Aitken. 2013. No. A09-1066. Supreme Court of Minnesota, United States of America.

People v. Susalla. 1974. 220 N.W.2d 405, 392 Mich. 387. Michigan Supreme Court, United States of America.

Perkins, RM. 1969. Perkins on Criminal Law, 2nd edition. Foundation Press.

Rouson v. Eicoff. 2006. 04-CV-2734 (ARR)(KAM). District Court of New York, United States of America.

Spear, VG. 2005. Leadership in Medieval English Nunneries .Woodbridge.

Wang, P. 2011. On Preservation Notarization of E-Contract. Proceedings of the 2011 International Conference on Management and Service Science (MASS), pp. 1-3.

White, CL. and Beaudry, MC. 2009. Artifacts and Personal Identity. In David Gaimster and Teresita Majewski (eds.) International Handbook of Historical Archaeology.