Project Milestone

Marcus A. Zimmermann
Marcus.Zimmermann1@Marist.edu

May 6, 2020

Abstract - As technology has progressed, it has become more reasonable to store patient information in the form of electronic medical records, allowing healthcare providers to leverage the profound speed and storage capabilities of emerging technologies. However, the transition to electronic storage poses new challenges to the long-standing practice of strict doctor-patient confidentiality. To keep patient information as private and secure as possible, it is standard to encrypt electronic medical records before storing them. This paper documents the design, development, and subsequent analysis of Healthcrypt, a mock storage system built to meet this standard.

Introduction

Healthcare providers have always taken extreme measures to ensure doctor-patient confidentiality, using systems like on-premise, lock-and-key file repositories to prevent the disclosure of personal information. However, as technology has progressed, it has become more reasonable to store this information in the form of electronic medical records. In other words, instead of maintaining a vast, tangible file system, traditional patient files have been systematically digitized and stored in electronic mediums to take advantage of the profound speed and storage capabilities of emerging technologies. These systems are fast, efficient, and scalable well beyond traditional storage mechanisms. However, they pose new, significant challenges to security as well.

The inherent connectedness of electronic storage systems broadly extends the attack landscape, putting sensitive information at risk of sophisticated, novel forms of theft [4]. To strengthen the defense posture of electronic storage systems, it is standard to use state-of-the-art encryption methodologies as part of the filing and storage process, mitigating the potential harm of inappropriate and unwarranted access to sensitive information.

The remainder of this paper is organized as follows: The Background and Motivation section introduces a brief history of healthcare security, as well as the motivation for the development of Healthcrypt; The Methodology section explains the design and development of Healthcrypt, including instructions to replicate the environment and launch the application for personal experimentation; the Experiments section describes Healthcrypt's performance on various tests of security; Lastly, the Discussion and Conclusions section describes observations made during the design, development, and analysis of Healthcrypt.

Background and Motivation

As early as the 1960s and 1970s, academic medical centers, government institutions, and private industry recognized the potential of electronic medical records and the "benefits to industry-wide standards," noting the need to create "organizations to tackle the broader issues that would facilitate the widespread use of electronic medical information" [1]. These broad-sweeping standards naturally included concerns related to the use of and access to electronic storage systems, as "increasing the connectivity capabilities also creates cybersecurity risks" like "unauthorized access to patient health information" and "changes to prescribed drug doses," not to mention many other causes for concern [4]. Unauthorized access to, and manipulation of, patient information is not something to be taken lightly.

The Security Rule under the Health Insurance Portability and Accountability Act (HIPAA) "establishes national standards to protect individuals' electronic personal health information that is created, received, used, or maintained by a covered entity. The Security Rule requires appropriate administrative, physical and technical safeguards to ensure the confidentiality, integrity, and security of electronic protected health information" [2]. That sounds comforting, but there is a minor detail pertaining to the implementation specifications of these technologies that may worry patients and healthcare practitioners alike. Each Rule contained within HIPAA is labeled as either 'Required' or 'Addressable.' What's the difference? Required Rules are self-explanatory. They are required. Addressable Rules, on the other hand, introduce opportunities to fall short of standards, including those related to security.

Addressable Rules, such as the practice of encrypting data at rest and data in transit, are necessary when "risk analysis shows such risk to be significant," the judgment for which is deferred to the covered entity [2]. In other words, implementing some form of encryption is left to the discretion of the health care provider. So, depending on the circumstances, your personal health information may not be encrypted at rest or in transit.

It is worth noting that the United States is among the leading countries when it comes to upholding encryption standards in healthcare [3]. This includes enterprise-level encryption strategies, which certainly offer some reassurance when it comes to the safety of patient records. However, we're not perfect. Not all data are encrypted, and data breaches are still well within the realm of possibility.

To better understand the systems storing and exchanging electronic medical records, the remainder of this paper documents the design, development, and analysis of Healthcrypt, an application that intends to offer a rudimentary exploration of encryption within the context of healthcare.

Methodology

Healthcrypt was built with Flask - a micro web framework - and a variety of extensions. Flask was chosen because it is extremely lightweight and provides the ability to seamlessly integrate extensions as if they were built into Flask already. In other words, "Flask aims to keep the core simple but extensible ... Flask can be everything you need and nothing you don't"[5]. For sample, proof-of-concept applications, few frameworks are better designed than Flask, making it a perfect candidate for the development of Healthcrypt. In the paragraphs that follow, we will discuss the configuration and routing mechanisms provided by Flask and implemented in Healthcrypt; the major extensions leveraged by Healthcrypt, including Flask-SQLAlchemy and Flask-WTF; and, lastly, Healthcrypt's encryption prior to storage, decryption prior to display, and the underlying application logic and extensions that enable these features.

First things first, Healthcrypt was built inside a virtual environment. Of course, this is standard procedure for many developers. Isolating development environments with venv - a virtual environment builder conveniently packaged with Python 3 - allows us to isolate our development environment and share projects with a reduced risk of compatibility issues. With the virtual environment generated and activated, we can begin installing packages fundamental to Healthcrypt, including Flask itself.

Flask provides default application structures to support larger applications with many extensions. More specifically, applications are built into packages defined by an "_ _init_ _.py" file.

Experiments

Section Pending Application Completion

Discussion and Conclusion

Section Pending Application Completion

References

[1] Jim Atherton. Development of the electronic health record, 2011.

[2] Office for Civil Rights. Hipaa for professionals, 2017.

[3] Ponemon Institue. 2020 global encryption trends study, 2020.

[4] NIST Information Technology Laboratory. 2017 annual report, 2017.

[5] The Pallets Projects. Foreword: What does “micro” mean?, 2010.