Edlira Martiri
Norwegian Biometrics Laboratory, Information Security Department, NTNU
Finse Winter School: Research topic presentation

Supervisors: Bian Yang, Christoph Busch

TOPIC: BIOMETRIC TEMPLATE PROTECTION: A DECEPTION-BASED MECHANISM

INTRODUCTION

From a security perspective, the protection of templates stored in the database of a biometric system is one of the main challenges. An adversary can try to carry out a modification of their contents or even an unauthorized transfer of templates from the database towards another system. To prevent and detect leakage of biometric information, in addition to a better control of database access, other techniques should be implemented. These improvements would not only prevent or slow down attacks but, even warn if such a leak has occurred.

While masquerade attacks are possible to cope with by better anti-spoofing technologies, they are not very possible to be completely prevented. To discourage both the physical and the digital masquerade attacks, we do this by empowering the system with detectability of protected templates leakage.

BIOMETRIC TEMPLATES AS HONEY OBJECTS

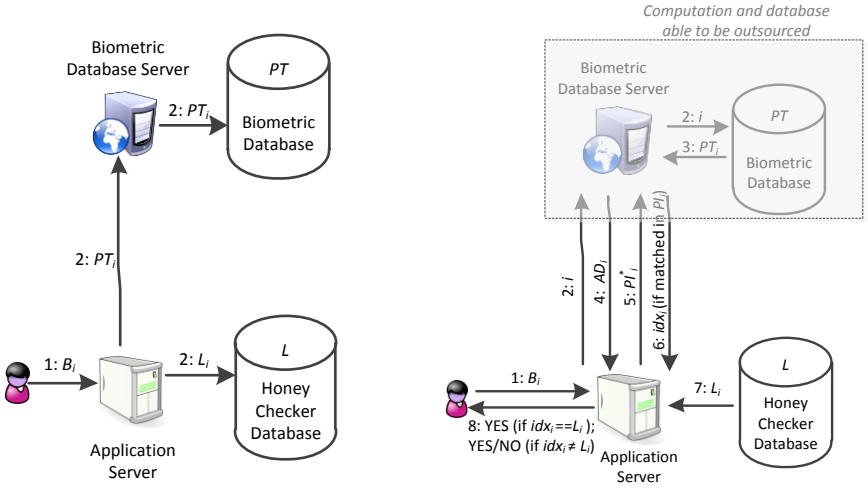
Honey objects are used in various aspects of system security to deceive internal threats or external intruders (be they people or machines) against unauthorized data access. An example are honeypots mostly used for the detection of outer intruders, which are network machines used to distract adversaries from other more important machines, and honey farms (a network of honeypots) [3] enabling deep research into server-side attacks. Subsequently, honeytokens [4] are mostly implemented against internal threats and another development we find at system level are honeyclients [5], the complementary of honeypots, designed to mimic the behavior of a web browser. On the data level we find solutions such as the honeywords and honeydocuments. The honeyword method [1] hides the password of a user between k hash values of random passwords, and honeydocuments [6] is again a trap-based mechanism which uses decoy documents. All these mechanisms serve as a safeguard against adversaries who try to get unauthorized data access. In [7] honey objects must comply to two main properties: (1) indistinguishability, honey and real objects must be hard to distinguish from each other (e.g. a real password from a generated password or a database entry of a real patient in a health system from a fake one); (2) secrecy, the real object must be secret and camouflaged among the honey objects. In the case of honeywords, if an intruder by some means gets access to the user' set of passwords, he can use/guess only one of them. The system intercepts that a honeyword is used, it will consider this as information leakage and proceed with further steps (set off an alarm and/or update the passwords set).

The honeywords method provides us a systematic way to counter the masquerade attack against protected biometric templates. It resorts to probability (i.e. information-theoretic security) instead of computational complexity based security to cope with the crackable-hash assumption. In the biometric context, most databases are facing the same challenges. In hash based biometric template protection scheme, such as fuzzy commitment [8], and secure sketch [9], if the hash is cracked, then the adversary can estimate the pre-image of the biometric features. And for feature-transformation based BTPSs (in [10] and [11]), the masquerade attack is even more straightforward. This is because the protected templates, PTs, are compared directly with a distance threshold and the attacker can find a PT's pre-image (biometric feature) with normally less effort than the case finding a pre-image of a hash value. As a result, for every enrolled user in the database, we need to provide a protection mechanism which needs to be applied on all the sweet templates (sugar and honey) and satisfies the abovementioned properties. Firstly, templates must be constructed in such a way that an adversary is not able to distinguish a sugar from a honey one, even if he: breaks the protection mechanisms; uses automatic tools such as classifiers; or tries to visually capture differences of honey and sugar templates pre-images to differentiate them. Secondly, the sugar template must be placed in a random position in the user database entry, or user data file, among the honey templates and this specific index must be known only to the honeychecker. We note that the aim of our approach on biometric templates, as well as the honeywords method, is not to lure the intruders with fake data, but to provide a means to alert the system that an internal or external adversary had access to the users' data and used them back: in other words that there have been system attack, information leakage, and user impersonation (masquerade attack).

Honey objects database design

The architecture design of a biometric system using honey templates is presented in fig. 1. During enrollment the

Application Server converts the plain biometric feature B_i of user i to a set of protected templates. This set is defined in the same way as we did with the protected objects PO, i.e. $PT_i = \{AD_i, PI_{i1}, PI_{i2}, \ldots, PI_{iK}\}$. PT_i. It will be stored in the Biometric Database and the index L_i in the Honey Checker Database. During verification the user i will show his biometric characteristic and the Application Server will retrieve the PI^*_i from the Biometric Database and send it to the Biometric Database Server. After comparing PI^*_i with all the PI_{ij} (j = 1, 2, …, K) of user i, it will send the best-matched template's index to the Application Server. If the index idx_i matches L_i the user will grant access to the system, otherwise an alarm will be set off and specific rules will follow, according to the defined security policies of the system. In the latter case, if the user personal identifiers match but the templates' indices do not, the system will consider this attempt as information leakage.



(a) Enrollment (b) Verification

Fig. 1. Architecture for a honey templates based biometric system [2].

This architecture can be applied to different biometric characteristic. Our first attempt is in the construction and protection of honey face templates. The main challenge is to generate honey objects, in our case synthetic face templates, which cannot be distinguished from real templates.

REFERENCES

[1] Juels, A.; Rivest, R.: Honeywords: making password-cracking detectable. In Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security (ACM-CCS'13), 2013, pp. 145-160.

[2] Yang, B.; Martiri, E. "Using Honey Templates to Augment Hash Based Biometric Template Protection". In proceedings of the 39th IEEE International Computers, Software & Applications Conference, 2015 (in press).

[3] Honeypots. http://www.honeypots.org/, accessed: May 2015.

[4] Spitzner, L.: Honeytokens: The other honeypot. In Symantec Security Focus, July 2003.

[5] Nazario, J.: Phoneyc: A virtual client honeypot. In Proceedings of the 2nd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more. USENIX Association, 2009.

[6] Bowen, B.M.; Hershkop, S.; Keromytis, A. D.; Stolfo, S. J.: Baiting inside attackers using decoy documents. In Security and Privacy in Communication Networks, pages 51–70, 2009.

[7] Juels, A.: A bodyguard of lies: the use of honey objects in information security. In Proceedings of the 19th ACM symposium on Access control models and technologies. ACM, 2014.

[8] Juels, A; Wattenberg, M.: A fuzzy commitment scheme. In CCS '99, ACM, 1999, pp. 28–36.

[9] Sutcu,Y.; Li, Q.; Memon, N.: Protecting biometric templates with sketch: theory and practice. In IEEE Transaction on Information Forensics and Security, 2(3), 2007, pp. 503-512.

[10] Ratha, N.; Connell, J.; Bolle, R.M: Enhancing security and privacy in biometrics-based authentication systems. In IBM Systems Journal, 40(3), 2001, pp. 614–634.'

[11] Teoh, A; Goh, A; Ngo, D.: Random multispace quantization as an analytic mechanism for biohashing of biometric and random identity inputs. In IEEE Trans. Pattern Anal. Mach. Intell., 28(12), 2006, pp.1892–1901.