



Privacy Preserving and Resilient Cloudified IoT Architecture to Support eHealth Systems

Jarkko Paavola^(✉) and Jani Ekqvist

Turku University of Applied Sciences, Joukhaisenkatu 3, Turku, Finland
{jarkko.paavola, jani.ekqvist}@turkuamk.fi

Abstract. Significant improvement in eHealth services in both quality and financial points of view are possible if public cloud infrastructures could be utilized in storing and processing personal health information (PHI) from IoT devices monitoring and collecting data from persons. The challenge is that personal health records are highly sensitive and health related organization are not willing to trust the cybersecurity of public clouds. Another challenge is that strict regulation is in place regarding the physical location of PHI. This paper addresses these issues by proposing tokenization architecture and crypto-implementation for personal identity number (PIN). This will allow the storage and processing of the personal health information (PII) in the public cloud as the data cannot be identified to a specific person. The proposal follows the general data protection regulation (GDPR) by offering secure and highly resilient architecture for the separation of health data and person identity.

Keywords: IoT · Cloud · eHealth · Information security · Resiliency
Privacy · Tokenization

1 Introduction

Industrial Internet consortium (IIC) envisions the emergence on industrial internet in five different sectors: energy & utilities, manufacturing, transportation, public sector, and *health care*. IIC estimates that the Industrial Internet can help globally drive down annual healthcare costs by roughly 25%, or about \$100 billion a year [1].

There are several visions what industrial Internet, or Internet-of-Things (IoT), could offer for health care. For example, Microsoft envisions transforming patient care with IoT [2], which could provide health organizations with the ability to leverage the cloud in new ways to transform their operations for reducing operational costs through predictive maintenance, real-time asset monitoring, or utilizing data across the entire continuum of care with analytics.

Cost-efficient global business operations require sophisticated utilization of cloud-based information system tools. Cloud computing is an approach to control the consumption of computing resources. It involves groups of remote servers and networks that allow centralized data storage and online access to computer services or resources. Clouds can be classified as public, private or hybrid. Another classification is whether the cloud is utilized as infrastructure, platform, or application. The integration of cloud computing and industrial Internet is an open research question [3].

The source of wide interest on cloud platforms for companies is to expand their businesses geographically around the globe and reduce the operational costs. In the long term, the approach will offer possibilities to utilize e.g. big data to offer novel services. Strategic intent for companies is to offer health care/medical software, products and platforms as a cloud service globally to different countries. Benefit is clear - the return of investment, using the envisioned approach when compared to the traditional approach, is much bigger.

Regulation sets strict demands for information systems processing health related data. The goal is to protect the rights of patients regarding information processing and archiving. Regulatory rules vary from country to country making exporting medical information systems complicated, since they must be tailored to each market and compatibility to regulatory demands must be proven with audit process. These different regulatory demands must be taken into account in very early stage of the development process to ensure that the adaptation to different regulatory demands does not require complex technical changes. This is especially true for new technologies, where legislation may be delayed.

Further, the General Data Protection Regulation GDPR [4] requires “appropriate technical and organisational measures to ensure the level of security appropriate to the risk” from all data processing related to Personally Identifiable Information (PII), and especially sensitive data as Personal Health Information (PHI).

Cloudified IoT and related security challenges has been investigated recently for eHealth applications e.g. in [5–7]. The current literature on the focuses mainly on security protocols for the communications. This paper has a different approach as the privacy will be provided by separating PII from PHI. The information security is protected by extremely resilient architecture, which is designed to minimize the attack surface and to fulfill GDRP requirements.

The work for this paper has been implemented in RAMP (Industrial Internet Reference Architecture for Medical Platform) project, where the approach has been to design open reference architecture, which allows to analyze all required components and services, and to create strategic roadmap for technology and IoT platform selections. A special case considered in the project is that the IoT medical device is located inside hospital. The data to be transferred out from the hospital may be categorized as follows:

- Data related to the conditions of equipment
- Sample data from the patient without identifying information e.g. cell sample for research purposes
- Patient data accompanied with identifying information

Here, we consider the latter case, which is by far the most problematic from the privacy point-of-view as PII and PHI must be separated. When public cloud is utilized, the physical location of the data, in principle, is not anymore under the control of the hospital that has produced the data. To minimize the risk of compromising personal health records in the case data breach within cloud provider this paper proposes

- Architecture for tokenization of personal ID
- Method of cryptographical operations to provide adequate level of security

The proposal has been verified in the real hospital environment pilot as described in Sect. 4.

As an example, this paper presents tokenization architecture and its crypto-implementation for Finnish personal identity number (PIN). This allows the separation of PHI from the PII in the cloud, thus enabling the utilization of public clouds for storing and analyzing patient data even from the hospital environment. The proposed solution is piloted with the hospital in southwest Finland.

The paper is organized as follows: the next Sect. 2 describes the proposed tokenization architecture and cryptographic implementation. Then, in Sect. 3, security analysis for the proposed architecture is presented. Section 4 describes the pilot implementation, and finally, conclusions are drawn in Sect. 5.

2 Tokenization and the Proposed System

Tokenization replaces sensitive information with a surrogate token that does not have any intrinsic value. The purpose of tokenization is to remove sensitive information from the data processing systems and store it in a single protected container.

Here we consider the case of protecting Finnish Personal Identity Number. The PIN is considered sensitive information according to the Finnish Personal Data Act [8] (available only in Finnish and Swedish), and by the General Data Protection Regulation [4]. In our case, it identifies the person whose Personal Health Information is being processed. The PIN is constructed from person's date of birth, a gender identifier, a random number selected from a space of 500, and a checksum character [9] (available only in Finnish and Swedish). A valid example of a PIN is 120188-479P, where '120188' is the date of birth, '-' is the century of birth, '479' is the random identifier (it being an odd number means a male), and 'P' is the checksum calculated over the preceding numbers. It is relatively trivial to find out date of birth and gender, which leaves us with a pool of entropy of only 500 numbers, or about 9 bits. Additionally, numbers 0, 1 and 900–999 are reserved. Therefore, there is a high risk of an attacker being able to reverse the tokenization process if keying is not considered carefully.

Tokenization is widely used in credit card industry for protecting credit card numbers. The proposed architecture follows the approach in PCI DSS standard [10]. However, methods commonly used for credit card number tokenization are not sufficient for storing PIN in the case of PHI. There are less than 10 million PINs currently in use and the whole valid value space from 1900 to 2017 is less than 43 million (the oldest person alive in Finland is 108 years old). If attacker gains access to the token database, he/she will be able to generate a rainbow table of possible PII values and reverse the encryption.

Therefore, we propose an architecture where encryption keys for each PII value are stored in separate server. This provides an added layer of protection without incurring the cost of provisioning a High Security Module (HSM). In addition to improved security, the system is also more resilient than the one in [10].

2.1 Tokenization Architecture

The proposed architecture follows strict separation of duties where a security breach of a single component does not compromise the overall integrity of the system. The architecture is illustrated in Fig. 1, where Processing Cloud denotes any public cloud used to process PHI.

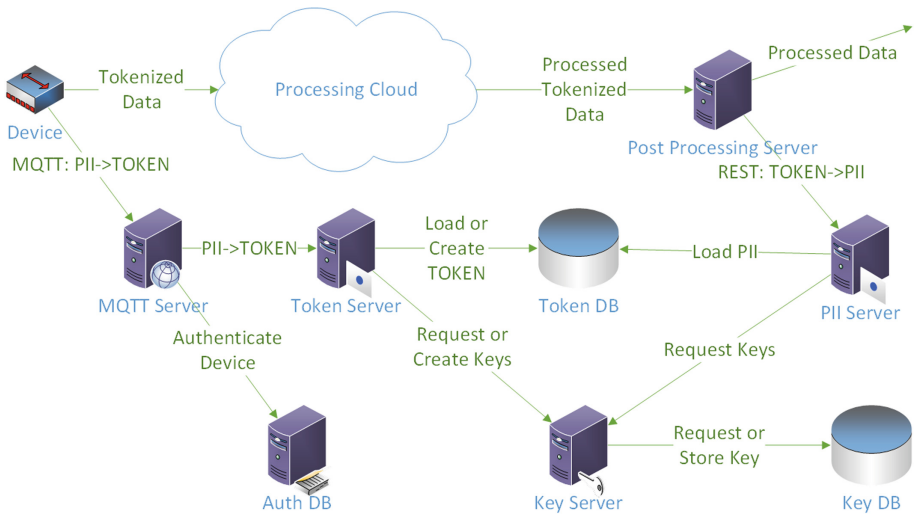


Fig. 1. The system architecture

Message Queue Telemetry Transport (MQTT) [11] was selected as the token query protocol, as it is a lightweight protocol widely used in IoT devices. It provides adequate security for data in transit as it can utilize TLS as transport protection. The architecture requires device authentication so that only trusted devices are able to communicate with a *MQTT server*. A *device Authentication database* is denoted with Auth DB in Fig. 1

The PII is stored encrypted in a *Token Database (Token DB)*. Encryption keys for the PII data are stored in a *Key Database (Key DB)*. When tokenization is requested from a *Token Server*, it queries encryption keys from a *Key Server*. The Key Server will return a key in constant time for any request to protect the server against timing based side channel attacks. If the key does not exist in the Key DB it will be created. Key mappings in the Key DB are protected using keyed HMAC, which derives the key index from partial PII and authorization keys stored at the Token Server and a *PII Server*. The PII server acts as de-tokenization server.

2.2 Cryptographic Procedure

In our implementation SHA-256 algorithm is used as the hash function for the HMAC construct and AES-256 is utilized in CBC (Cipher Block Chaining) mode of operation for the PII data encryption. Keying is done in four steps as illustrated in Fig. 2, where Requester is the IoT device:

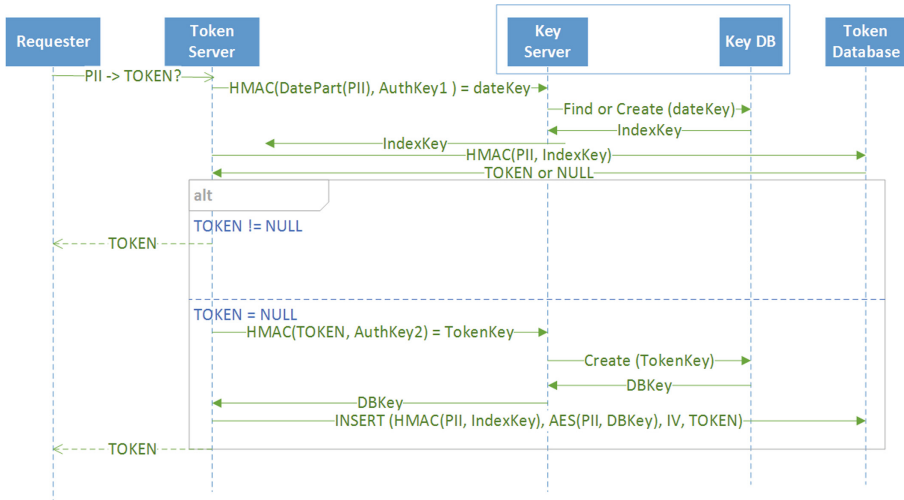


Fig. 2. The keying process

1. Token Server creates Authorization Key 1 ($AuthKey1$) to generate a HMAC construct from the date part of PII data: $HMAC (DATEPART (PII), AuthKey1)$. The authorization key is stored to the Token server and to the PII server.
2. This construct is used to query the Key Server, which will return an existing key (denoted as $IndexKey$) or generates a new one if this was the first request for the given HMAC. In this way, there will be 36525 valid keys in the key database, and thus the Token Database is divided into 36525 separately encrypted partitions. The $IndexKey$ is then used to create an index query to the Token DB by utilizing the HMAC construct again, this time with the whole PII data: $HMAC (PII, IndexKey)$.
3. In the third step, assuming the PII data does not exist in the database yet, the Token Server will use a separate Authorization Key 2 ($AuthKey2$) to generate another key request to the Key Server. This HMAC construct hashes the newly generated token into a key identifier: $HMAC (Token, AuthKey2)$. The Key Server will return another key that we denote a $DBKey$. These keys are stored alongside the $IndexKeys$ in the Key DB and are indistinguishable from each other.
4. The $DBKey$ is finally used to encrypt the PII data in the Token Database. This allows the detokenization server (PII Server) to acquire decryption keys for the PII data utilizing only the Token value and $AuthKey2$. The tuple in the Token DB is thus $(HMAC (PII, IndexKey), ENC_{AES}(PII, DBKey, IV), IV, Token)$. Here, the $HMAC (PII, IndexKey)$ is used as an index to allow retrieval of the Token as described previously, $Token$ is the generated Token in plain text to allow the PII Server to index the database, $ENC_{AES}(PII, DBKey, IV)$ is the AES encrypted PII data, and the initialization vector IV is used to allow tokenization of data exceeding the cipher block size.

The system can be modified to provide additional transport security in case the transport from device to the Token Server cannot be completely trusted. Then, `AuthKey1` is distributed to devices, and they calculate the HMAC to access `IndexKey`. Now, the Token Server can use this HMAC construct to directly query for the `IndexKey`, and return the Token to the device if it is found. Only if the Token does not exist in the system yet, is the PII data requested from the device. This reduces the window of sensitive data exposure considerably. In the case of small dataset like the Finnish Personal Identity Number, all the Token and Key data can be pre-generated in to the databases. This mode of operation is however susceptible to `AuthKey1` leak.

3 Security Analysis of the Proposed System

The General Data Protection Regulation requires “appropriate technical and organisational measures to ensure the level of security appropriate to the risk”. The regulation explicitly mentions pseudonymization and encryption of personal data as an advisable technical measure [4]. Our system provides technical measures that fulfils and exceed the requirements for ensuring the security of the PII data during the processing. The security of the system is built on several layers. The requirements of the server architecture follow industry standard best practices such as the PCI DSS Tokenization Guidelines [10] and Cloud Security Alliance Security Guidance [12]. Cryptographic algorithms are selected based on ENISA [13] and NIST [14] guidance. The security of the system requires that the servers and network are configured to allow only authorized access.

3.1 Attack Surface

The tokenization system provides high resilience against intrusions. High emphasis is given on attack surface reduction. The system only has three points of entry for legitimate and illegitimate access: the IoT device, the MQTT Message Server and the Cloud Control Interface. The Cloud platform and control interface are assumed to be protected by industry standard measures, which are outside the scope of this document. The IoT device is the hardest component to protect as it resides outside our control in the customer network. Therefore, material sensitive to the security cannot be stored in the device, and the device must not provide any open services. If attacker has access to the device, he/she will be able collect the PII data at the device input, and thus we cannot protect against it. The protection of the device is manufacturer’s and device operator’s responsibility. The MQTT Server is the only component that is reachable through Internet. It exposes only the secure-mqtt (TCP 8883) port and service.

3.2 Access Security

Connections to the MQTT Server must be authenticated and verified to be from devices authorized to access the Token Server. All other components of the service must allow connections only from authenticated and authorized components of the system as shown in Fig. 1. The connections between components of the service must be secured

using TLS. The network of the Token Service must be separate from other networks, e.g. a separate virtual private cloud in cloud service, and it must only contain servers described in the architecture.

MQTT standard provides only plain username and password based authentication, although the connection is secured using TLS. Device authenticity is enhanced by utilizing an external authentication provider. This allows the system to provision per device credentials automatically instead of using shared key. Passwords are replaced with a key stored using key derivation function. Here, PBKDF2 (Password-Based Key Derivation Function 2) method is selected and parametrized following [12–14].

3.3 Architecture and Cryptographic Resiliency

The security of the PII data in the system is based on standard cryptographic constructs and the strict separation of keys from the protected data. The protected data is stored only in the Token Database and the keys necessary to decrypt it are in the separate Key DB. The authorization keys necessary to access the correct keys in the Key DB are stored in the Token Server and the PII Server, so when data is at rest, unencrypted access to it requires access to three separate servers. When data is in use, it is accessible in the memory of the using server. Thus, the memory must be overwritten as soon as processing ends. It should be noted that as we form the first key indexing value based on the date of birth, attacker is also able to construct this value for retrieving their key if he has access to `AuthKey1` and is targeting a known person. The Key Server will return keys only in constant time for any request to protect the server against side channel attacks.

The cryptographic resilience of the system is also based on layered approach. The tokens are generated using secure random number generator, which does not take the PII data as an input and can thus be considered completely separate. The attacker cannot gain any insight into the PII data from the token itself. If attacker has access to the Token Database and can extract the data, he/she has to consider the values of a tuple individually and as a combination. The PII data itself is encrypted using AES-256 in CBC mode using a random initialization vector. The best known attack against the AES-256 can currently recover the key with time complexity of $2^{254.27}$ and data complexity 2^{40} [15], so the algorithm can be considered secure. The Token Database index is created from the PII data using HMAC construct with SHA-256. HMAC security is based on the hash algorithm's resistance against preimage attack and the length of the key used, up to the length of the hash algorithm output. Best known preimage attacks against SHA-256 can discover the input data only on 45 reduced round variant with time complexity of 2^{255} [16]. As long as authentication keys contain at least 256 bits of entropy the algorithm can be considered secure.

Final issue is that the tuple contains the PII data in two forms, encrypted and hashed. There are no known attacks that could utilize a hashed value as a known value to break a symmetric encryption. In our system, the value is further obstructed by the HMAC construct.

4 Architecture and Crypto-System Verification in the Real Hospital Environment Pilot

For the pilot, a device simulating the operation of PerkinElmer GSP product was installed in the medical network of hospital in southwest Finland. The medical network is operated by company providing IT services for the hospital. The pilot implementation was approved in the information security process of the hospital.

The implementation architecture is presented in Fig. 3, which follows the architecture shown in Fig. 1. Device producing the data is located in the hospital network. The outer firewall of the hospital networks has ports open required to access Microsoft Azure and Amazon Web Services (AWS) public clouds. Tokenization operations according the Sect. 3 are implemented in the AWS. However, these operations could be performed in any network e.g. company private network. The main issue is that tokenization operations and data processing are separated to different networks. For data processing Azure cloud is used. IoT Hub is the component responsible for communication with the device in JSON format, and is receiving the data. The data samples are processed with Stream Analytics, stored to Document DB, and visualized with Power BI.

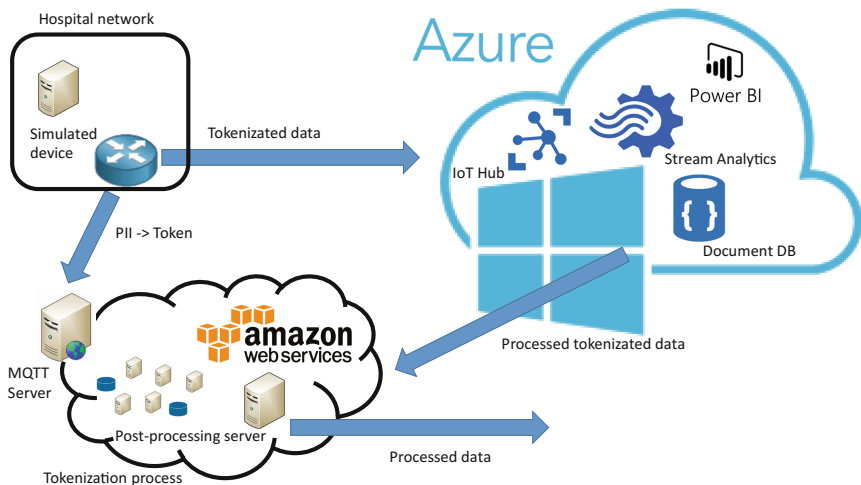


Fig. 3. Pilot implementation structure

A sample of the simulated data in JSON format generated by the device compared to the tokenized data ready for processing in the public cloud is shown in Fig. 4. The figure shows that the PIN value has been changed to the token value.

As the token is static, in addition to comparing single measurements against preset limits, the processing can perform a time-series analysis based on historical data of the same patient. The processing methodology and results are outside the scope of this document. When processing is finished, the data is passed to post-processing server.

This server will query the PII Server of our tokenization system with the token, and replace the token with actual PII data to allow the identification of the patient. The processed data is then forwarded into clinical database for diagnosis.

```

"SpecimenID": "BC123456",
  "Info": {
    "PIN": "120188-479P",
    "Tests": [{
      "Test": "hCGb",
      "Concentration": "78.56",
      "Unit": "ng/ml"
    },
    {
      "Test": "PAPP-A",
      "Concentration": "784.3",
      "Unit": "U/ml"
    }
  ]
}
}

"SpecimenID": "BC123456",
  "Info": {
    "PIN": "oxUKFYC36RdssRE",
    "Tests": [{
      "Test": "hCGb",
      "Concentration": "78.56",
      "Unit": "ng/ml"
    },
    {
      "Test": "PAPP-A",
      "Concentration": "784.3",
      "Unit": "U/ml"
    }
  ]
}
}

```

Fig. 4. Simulated device analysis data and tokenized analysis data compared in JSON format

5 Conclusions

Significant improvement in eHealth services in both quality and financial points of view are possible if public cloud infrastructures and their inherent advantages with global data centers and calculation power in big data analytics can be utilized. However, privacy concerns and especially requirements from the GDPR set difficult challenges to cloudified IoT architectures for eHealth.

This paper addressed these issues by proposing tokenization architecture and crypto-implementation for PII. The example implementation was Finnish PIN. The proposed system will allow the storage and processing of the PHI in the public cloud as the data cannot be identified to a specific person. The proposal follows general data protection regulation (GDPR) by offering secure and highly resilient architecture for the separation of health data and person identity.

The proposal was shown to minimize the attack surface, provide access security as suggested in [12–14], and to provide resilient architecture and cryptographic implementation with layered approach. The system has strict separation of duties where a security breach of a single component does not compromise the overall integrity of the system.

Acknowledgement. This work was supported in part by the Finnish Funding Agency for Innovation (TEKES) under the project “Industrial Internet Reference Architecture for Medical Platforms” (RAMP). The project is partly funded by industry partners Wallac Ltd/PerkinElmer Inc, Etteplan Oyj, Atostek Ltd, and Nextfour Group Ltd.

References

1. Industrial Internet Consortium. <http://www.iiconsortium.org/>
2. Microsoft. <https://www.microsoft.com/en-us/internet-of-things/healthcare>
3. Botta, A., de Donato, W., Persico, V., Pescapé, A.: On the integration of cloud computing and Internet of Things. In: 2014 International Conference on Future Internet of Things and Cloud (FiCloud), pp. 23–30. IEEE Press, New York (2014)
4. General Data Protection Regulation (EU) 2016/679. <http://eur-lex.europa.eu/legal-content/en/TXT/?uri=CELEX%3A32016R0679>
5. Ida, I.B., Jemai, A., Loukil, A.: A survey on security of IoT in the context of eHealth and clouds. In: 11th International Design & Test Symposium (IDT), pp. 25–30. IEEE Press, New York (2016)
6. Supriya, S., Padaki, S.: Data security and privacy challenges in adopting solutions for IOT. In: 2016 IEEE International Conference on Internet of Things (iThings), pp. 410–415. IEEE Press, New York (2016)
7. Sawand, A., Djahel, S., Zhang, Z., Naït-Abdesselam, F.: Multidisciplinary approaches to achieving efficient and trustworthy eHealth monitoring systems. In: IEEE/CIC ICC3 2014 Symposium on Privacy and Security in Communications, pp. 187–192. IEEE Press, New York (2014)
8. Henkilötietolaki 523/1999. <http://www.finlex.fi/fi/laki/ajantasa/1999/19990523>
9. Valtioneuvoston asetus väestötietojärjestelmästä 25.2.2010/128. <http://www.finlex.fi/fi/laki/ajantasa/2010/20100128>
10. PCI Data Security Standard Information Supplement: PCI DSS Tokenization Guidelines. https://www.pcisecuritystandards.org/documents/Tokenization_Guidelines_Info_Supplement.pdf
11. ISO/IEC PRF 20922. <https://www.iso.org/standard/69466.html>
12. Cloud Security Alliance, Security Guidance for Critical Areas of Focus in Cloud Computing. <https://downloads.cloudsecurityalliance.org/assets/research/security-guidance/csaguide.v3.0.pdf>
13. ENISA: Recommended cryptographic measures – securing personal data. <https://www.enisa.europa.eu/publications/recommended-cryptographic-measures-securing-personal-data>
14. NIST: Special Publication 800-175B: Guideline for Using Cryptographic Standards in Federal Government: Cryptographic Mechanisms. <https://www.nist.gov/publications/guideline-using-cryptographic-standards-federal-government-cryptographic-mechanisms>
15. Tao, B., Wu, H.: Improving the biclique cryptanalysis of AES. In: Foo, E., Stebila, D. (eds.) ACISP 2015. LNCS, vol. 9144, pp. 39–56. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19962-7_3
16. Khovratovich, D., Rechberger, C., Savelieva, A.: Bicliques for preimages: attacks on Skein-512 and the SHA-2 family. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 244–263. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34047-5_15