Technical feasibility of context-aware passive payment authorization for physical points of sale

Adam Wójtowicz¹ · Jacek Chmielewski¹

Published online: 17 June 2017

© The Author(s) 2017. This article is an open access publication

Abstract In this work, the technical feasibility of passive secure payments for brick-and-mortar points of sale is analyzed. The core element of the proposed approach is a new application for context-based risk and trust assessment. It allows for dynamic selection of payment authorization methods that constitutes accurate trade-off between security and convenience. Particularly, the payments can be performed and authorized in the background using biometric means (face recognition), without user's explicit action. Generally, in the proposed approach, multiple devices are used for authorization: mobile, wearables, or stationary, client's or seller's, and multiple authorization methods are used: biometric, knowledge-based, and possession-based. The reported research includes requirement identification, novel architecture and protocol proposition, proof-of-concept prototype system deployment, and evaluation-based lessons learned. The research confirms that with the proposed approach, it is possible to take advantage of client-seller trust dynamism to simplify the payment process while maintaining the security level.

Keywords Payment authorization · Passive payments · Deviceless payments · Context-aware authorization · Authorization rules · Biometric identification

1 Introduction

In developed countries, payments in brick-and-mortar retail stores are done mostly with smart cards. The most convenient smart card payment method is contactless authorization. However, this payment method has its drawbacks. First, it can be used only for small amounts; otherwise, knowledge-based authorization is required. Second, the process still involves a number of steps: the seller indicates the ability to pay using a payment terminal, the client finds her bag or wallet, the client draws the card, the client uses the card on the terminal, and the client puts the card away. Since each of these actions requires certain time and active commitment from the client, such a procedure can be inconvenient in many practical scenarios.

In our research, we aim to automate the payment process as much as possible, but not too much. The idea is to dynamically switch between multiple devices (mobile or stationary, client's or seller's) to simplify the identification and authorization process while maintaining the necessary security level. We take advantage of context-based risk and trust assessment in order to dynamically determine the optimal trade-off between security and convenience. We focus especially on routine, *recurring transactions*, where such recurring transaction is defined as a payment performed by a client multiple times in similar timespans (e.g., daily, specific days of a week), in the same place (particularly, authorizing multiple small orders during single visit), with the same vendor (but at a different location), or for a similar amount. Common examples of such payment patterns are morning coffee bought every day in the same café on the way to work, quick lunch at a self-service restaurant near the office, or a few times a month, the fuel payment at a neighbor gas station. Despite the fact that the recurring transactions usually involve small amounts, there are also medium- or high-amount ones, so that they constitute

✉ Adam Wójtowicz
awojtow@kti.ue.poznan.pl

¹ Department of Information Technology, Poznań University of Economics and Business, al. Niepodległości 10, 61-875 Poznań, Poland

a larger set of operations than those that could be conveniently handled using a contactless smart card.

The simplification of the payment process concerns reducing its execution time and minimizing the number of operations required to be performed by the client. Analyzing the sequence of actions required by a traditional contactless payment procedure, it is clear that simplification of this process is possible only by eliminating the need to authorize the payment by the client. In such scenario, the client orders products/services, receives them, and leaves without having to search for the wallet and pulling out her payment card or any other attribute necessary to authorize the payment. Payments made in this way are referred to as *passive payments* in this article. If a process of passive payment authorization does not require communication with user's device, it is referred to as *passive deviceless payment*. In the proposed concept, the decision how to authorize the payment (passively or actively; with biometrics, possession, or knowledge; choice of a specific method from these classes; choice of the device) is done by a payment operator (e.g., a bank, a payment platform) on behalf of the client.

In this paper, requirement identification is followed by a proposal of a system architecture that enables passive deviceless payments, a listing of technologies which could be used to build such a system, and conclusions from a proof-of-concept prototype deployment. The construction of such a system requires defining a sequence of operations that eliminates payment authorization on the client side and allows for making it on the payment operator side. Therefore, it is necessary to introduce a phase of user identification, performed by the operator and transparent for the client, i.e., passive, to avoid offsetting the benefits of payment process automation.

In parallel to the goal of payment process automation, the developed system must meet expectations set for IT systems used in the financial domain. In particular, it must allow the payments operator to maintain control over the level of payment security. In a minimum variant, the ability to control payment security should be at least at the same level as for contactless payment cards. However, optimally, this control should be capable of adjusting the security level independently for each transaction—in the context of customer profile, trust in the seller, and other criteria.

Regardless of the functional requirements, it is important to ensure practical applicability of the developed solution and take into account existing alternatives. It can be assumed that market adoption is difficult if the new solution does not offer anything more than just technical innovations. Therefore, another requirement for the developed system is to provide all the actors of the payment process: the client, the seller, and the operator of the payment system with appropriate added value.

The assumed goals of this work are the following:

- To validate specification of a system that enables context-aware passive payments for physical point of sale (PoS)
- To prove added value provided by the system to each of the three actors: the client, the seller, and the payment operator

2 Background

For a few years, NFC-based contactless payments for EMV smart cards have become de facto standard [1] for low-risk transaction authorization in brick-and-mortar retail stores. Nowadays, this technology migrates from smartcard to mobile device as a carrier. Mobile services and application such as Apple Pay [2], Samsung Pay [3], or Android Pay [4] have been proposed to allow smartphone users for transaction authorization with their devices. Apple Pay supports also wearables (e.g., Apple Watch) and works even without constant pairing of the wearable with the smartphone.

As for the authorization, all these approaches are based on tokenization [5] approach. However, they differ with respect to the storage model of security-sensitive data that are used in the process of authorization. Apple's and Samsung's solutions, as they are limited to their vendors' hardware, can rely on the presence of embedded Secure Hardware Element on the device. Since Google does not control the hardware environment of the Android OS, it utilizes a software-based technology called Host-based Card Emulation. Therefore, in Android Pay model, the secret keys have to be stored in the cloud on Google's servers (as opposed to secure client-side chip), which requires Internet connectivity. Android Pay addresses the problem of user's offline periods by allowing to store a limited number of pre-generated one-time tokens on the device.

For user-side transaction authorization, in Apple Pay and Samsung Pay approaches, the relatively convenient and fast fingerprint biometrics can be used. This is possible, as in the case of secure key storage, because of the support from the side of device vendor, integrating and providing API for built-in fingerprint scanners. Such support is absent in the case of Google's solution.

Therefore, these solutions are either hardware vendor dependent and relatively convenient or hardware vendor independent and less convenient as well as less secure. Moreover, it has to be stressed that common general limitation of all these solutions is the requirement for a user to get the device out, carry it in hand, and move it close to the NFC sensor at the time and place of the transaction authorization. As in the case of a regular smart card, the user has to manually manipulate the carrier during the payment process.

To address this problem, Google Hands Free [6] approach has been proposed by Google, independently from Google's Android Pay. The Hands Free application uses Bluetooth Low Energy (BLE), WiFi, location services, and other sensors on the user's device to detect user presence near service provider. This enables the user to pay hands free, without getting out the smartphone or opening the application. During the transaction, a verbal declaration of participation is required from the user (the system is not fully passive) and identity verification with initials and profile photo from the cashier. Google is also running early experiments using automated facial identification to further simplify the checkout process with an in-store camera.

The payment system that is based on facial recognition to a larger extent, called Zero-Effort Payments (ZEP), has been proposed recently in [7], where authors interestingly discuss system evaluation results. The face identification results are promising, but the recognition is human assisted, i.e., a ranking of five most probable identities is presented to the human operator and he/she manually chooses the right one. The recognition process has demanded a heavy computational load since a number of faces are tracked simultaneously. Also, authors point out the low face recognition accuracy without supporting localization device. In their system, BLE localization devices have been used to significantly reduce the face recognition error rate. Therefore, it must be noticed that both Google Hands Free and ZEP are not deviceless systems, i.e., although a user does not need to manipulate her device during the payment process, she needs to carry a switched on device during identification and customer service.

Nowadays, also Uniqul system [8] is being deployed to provide fully deviceless payments authorized with user face image. However, it requires the user to type a PIN number when face identification has a low confidence level or tap the confirmation button on the in-shop tablet in the opposite case. Therefore, it cannot be considered as a fully passive approach from the end-user perspective.

There are also payment systems based on face recognition that utilize end-user smartphone camera. MasterCard has proposed a simple-to-use mobile solution [9] that allows customers to authenticate their online purchases using their own face images. It refers to the selfie phenomenon, which is natural for a number of end-users (digital natives). The application verifies image authenticity by detecting eyes blinking during image acquisition. MasterCard's approach is designed for online shopping, not brick-and-mortar trade. Contrarily, Lucova, using BLE technology proposes a system called FreshX [10] that also is based on selfie face image authorization but is dedicated for brick-and-mortar cafeterias. Such approaches, although natural, are neither deviceless nor passive, and security concerns can be serious.

There is a number of research and industry effort related to seamless payments focused on biometrics other than face,

e.g., fingerprints, or palm recognition. For instance, Liquid Pay [11] identifies customers by their fingerprints and, for extra security, by veins and electrical signals emitted by the human body. It has been installed in restaurants, fitness clubs, and theme parks. Payment systems based on the Fujitsu PalmSecure technology [12] recognizing vein patterns in a whole palm are being tested by [13] or [14] in many cafeterias. However, those technologies, although deviceless, stable, and relatively mature, cannot be perceived as passive.

All the mentioned solutions could be classified as either passive or active and either deviceless or device-based (Table 1). Most of the payment solutions are active and device-based. In some systems, biometric attributes are used instead of client's device to make them deviceless, but there is no solution that could be classified as both deviceless and passive. Our research shows how to migrate from the current state of the art towards deviceless and passive payments using context-awareness and dynamic risk and trust assessment as drivers.

3 Requirements for passive payment system

3.1 Payment automation

The idea of payments automation involves freeing the customer from the necessity of performing any active operations related to the payment and transferring payment process decisions and handling to the side of payments operator. Obviously, this should not deprive the customer of the control over her own resources; thus, the automation should require prior approval from the customer. This aspect is not included in this analysis as it is assumed that all customers served by the proposed system have given an informed consent to enable the payment automation.

In practice, payments automation requires close cooperation between the seller's and the payments operator's systems. The seller's system must inform the operator's system to initiate the payment (whom and how much to charge) and the operator's system must properly inform the seller's system of acceptance or rejection of the payment. Moreover, since the seller's system must provide information who should be charged, it must first identify the customer. There are two variants of possible seller-operator cooperation.

Table 1 Classification of payment solutions

	Active	Passive
Device-based	NFC-based contactless payment, Apple Pay, Samsung Pay, Android Pay, MasterCard Selfie Pay, Lucova FreshX	Google Hands Free, ZEP
Deviceless	Uniqul, Liquid Pay, Fujitsu PalmSecure	(this work's area of interest)

In the first one, the seller uses its own customer identities (e.g., from its own loyalty system). The operator must be informed beforehand about customer identity used in the seller's system and about its relation with customer identity used in the operator's system. Such information should be provided to the operator's system directly by a customer because it is the only actor of the process that has knowledge of both identities and can bind them.

In the second variant, the seller uses customer identity given by the operator. The identification mechanism working at the seller side must be informed about the identifiers of all possible customers in advance or must be continuously connected to the operator's system. In the latter case, it is possible to transfer the customer identification procedure to the side of the payments operator.

3.2 Passive customer identification

3.2.1 Assumptions

The introduction of payment automation is aimed at facilitating the process of payment for the customer, among others, by minimizing the number of actions that a customer must perform to authorize the payment. But the assumption that the seller's system identifies the customer could increase the number of these actions—e.g., an effort related to loyalty card validation is comparable to the contactless payment procedure. Since that could diminish the potential advantage, it is necessary to apply passive customer identification.

Passive identification is based on detection of the presence of a particular person at a particular location. The detection relies on what the person has (an object), or who she is (biometrics). The third option, based on what the person knows (a knowledge), is not applicable here because it always requires an active participation of the identified person. The passive identification based on objects could be implemented, for example, by the use of radio identifiers (Bluetooth beacon technology). However, this approach assumes that the person identified will always have to carry a relevant object. The passive method of identification which seems to be the most convenient in the scenarios under consideration is therefore biometrics. In turn, face recognition is the biometric method that can be implemented effectively without the active participation of the identified person [15].

3.2.2 Deployment requirements

For face recognition service, it is necessary to equip the seller location with appropriate infrastructure and to register customer face images in a database. The database can be maintained by either the seller or the payments operator. The database maintained by the seller could contain data of all customers of a given seller and handle all its PoSes, but if a similar

system would be used by other sellers, the client would be forced to go through the face biometric registration process with each seller individually. A solution less burdensome for the customer is a one-time registration of face biometrics handled by the payments operator, which could offer the customer identification service for a number of sellers. Moreover, it appears that this variant is easier to adopt because of the higher level of trust that customers have for payments operators than for sellers.

3.3 Control of payment security level

The postulated full payment automation is designed for routine payments—i.e., payments that meet certain requirements. These requirements include, for example:

- Accordance of the payment characteristic with a payment pattern built on the basis of the earlier payments of the user
- A certain level of confidence in the seller, or the seller point
- A certain level of risk dependent on the type of product/service or payment

In practice, the scope and characteristics of applicable requirements may vary for various payments operators—depending on their expectations and the data they are able to collect. What is more, it would be incorrect to permanently define the thresholds for such requirements, because they may vary for each customer–seller pair and may change over time. Consequently, it is necessary to use a mechanism that will dynamically evaluate the parameter values for a particular payment and based on an extensible set of rules will determine whether the payment can be classified as a routine payment or not.

4 Design of context-aware passive payment system

4.1 Data flow related to transaction procedure

To address the described requirements, a transaction procedure shown in Fig. 1 is proposed in this work. It is a complete picture of the information exchange between actors of the process—regardless of the form and the medium used to transfer this information. The procedure consists of two steps that need to be consecutive, but not necessarily in the same block of time: the client registration phase and the client service phase, which includes the payment process.

4.1.1 Client registration phase

In order to join the payment automation program, the client must already have a payment account at the payments

Fig. 1 Data flow of transaction procedure

operator (denoted herein as Operator) and then submit the following information:

- Face biometrics—used for passive customer identification at PoSes.
- Information about client's device—including data describing supported authentication methods. It is used to select authorization method for payments other than routine. In practice, the registration of this data can be automated: the necessary customer actions is limited to installing a mobile application provided by the Operator.
- Optionally, a list of client's identifiers assigned to her by retailers (e.g., a number of a loyalty card)—used to personalize the service by the seller, but not necessary for payment automation.

4.1.2 Client service phase

Operator client identity The seller obtains *Operator Client ID*—i.e., the client's identifier assigned in the system of Operator, or data stream that can be used to passively identify

the client (face images). The identifier can take any form (e.g., a number saved on client's membership card, RFID key fob label, a fingerprint).

Client identification request The seller's system sends the question regarding an identity of the client to the Operator. If the customer has registered her identifier assigned by the seller in the Operator's system, the identifier is returned to the seller, so that the seller can identify the client in his sales/loyalty system (Seller Client ID). If a client has not made such a registration, the seller receives back the same ID that has been provided by the client and the seller cannot personalize the customer service.

Personalized offer The seller presents the offer to the client. When using the Seller Client ID, the offer can be personalized.

Selected products or services The client selects products/services and places an order.

Debit request with the total amount The seller's system transmits a payment request to the Operator (including client identifier and the order amount).

The Operator decides whether the payment may be considered routine. If so, an automatic authorization is made and both parties (client and seller) are notified of successful completion of the payment. Thus, the client can pick up the goods or receive a service and leave. If it is not a routine payment, the Operator performs further actions.

Order summary and authorization requirements If the Operator does not classify the payment as routine, the client receives a request to debit her account with information about authorization methods that can be used to authorize the payment.

Authorization data The client accepts the debit request, chooses the most convenient authorization method that is allowed (manually or automatically using preferences rules) and performs authorization. The customer can also choose not to accept the debit request, which causes a notification sent to both parties (client and seller) about the failure of the payment.

Payment confirmation The Operator, after confirming the correctness of the authorization, provides both parties with a notification about successful completion of the payment. The customer can pick up the goods or receive a service and leave. In the case of payment authorization failure, the Operator immediately notifies both parties (client and seller) about the failure of payment, or it retries the authorization by sending the customer the next set of authorization methods to choose from.

Therefore, the customer service procedure is changed from a traditional sequence: order (O1) followed by a payment (P1), into a sequence: identification (I2), order (O2), payment (P2). Assuming stages O1 and O2 as comparable in terms of execution time, the goal is to get the total duration of I2 and P2 less than the time of P1.

4.2 Architecture of context-aware passive payment system

The system components are localized both at the client side and at the payments operator side and, to some extent, also at the seller side. On the client side BYOD model is assumed, so here, only software that integrates with client devices is required. At the seller side, hardware–software solution has been designed enabling the identification of clients and the use of an API for integration with sales/loyalty system of the seller. At the payments operator side (that can be perceived as a “cloud” service from client or seller point of view), there is a set of software modules that implement the main elements of the system logic. It is assumed that the software is running on the infrastructure maintained by the payments operator and is

available remotely through a secured communication channel. A detailed diagram of all the components of the system is presented in Fig. 2.

App component The *App* component is a mobile application build in native or hybrid technology that handles system notifications and utilizes hardware features of mobile devices, such as fingerprint reader. In practice, this element can be easily integrated with a regular mobile application of the payments operator. *App* component allows for registering the client's device in the system of the payments operator, along with information about authorization methods supported by the device (e.g., geolocation, entering a code on the keyboard, performing a gesture on the touch screen, executing a gesture in space, scanning a fingerprint, recording a voice). Then, it is responsible for receiving and presenting notifications concerning authorization of a non-routine payment or the payment processing status (e.g., payment failure due to lack of funds). Finally, it supports payment authorization methods and transferring the credentials of the selected authorization method to the payments operator. Registration of a device and supported authentication methods and the transfer of authorization results is carried out using API calls provided by components at the payments operator side. Receiving a request to authorize a non-routine payment is performed with PUSH notifications initiated by a component at the payments operator side.

Cam component *Cam* is a software-hardware component responsible for tracking, acquiring, segmenting, filtering, and streaming user face images at the PoS. The images are streamed to the payments operator side, to the *Ident* component.

Sell component Sales/loyalty software (*Sell*) receives PUSH notifications with information about customer identification and payment status and communicates with remote components located at the operator side, using API calls. It is assumed that the *Sell* is a third-party software and as such is not a part of the system—the system design includes only the API.

Ident component Client identification subsystem, which uses client face images sent from the PoS and patterns previously registered in the payments operator's biometric database. This component uses a permanent connection with *Cam* modules that work at PoSes and continuously processes the streams of images received from these modules.

For passive identification, face recognition based on a single image has a number of drawbacks. Apart from the risk of false matchings that would not be corrected automatically, such approach would require an additional effort from the seller side (“taking a photo” manually) and would require

Fig. 2 Architecture of context-aware passive payment system

active unnatural face presentation from the user, which is not compliant with the requirements described in Section 3.2. However, it can be assumed that there is a short but continuous time period in which a user prepares for the transaction (e.g., walks over, looks through the offer). This few-second period can produce several dozen face images, and then the initial identification could be performed. As an element of the proposed system, a rule-based heuristic algorithm has been developed in which final identification decision is a result of a number of face matchings calculated within the given time period. Therefore, a low number of false matchings do not impact the final identification decision. For single matchings any face recognition algorithm can be used, e.g., standard Eigenfaces [16].

ID component Client identities repository, which binds the user identities assigned by the operator and user identities used in systems of individual sellers. For performance reasons, the biometric identifiers are stored in the *Ident* component. The content of the repository is available for other components of the payments operator system through an API.

Devs component Registry of client's devices and available authorization methods providing an API that allows instances of the *App* component to register new data and other components of the payments operator system to access the registered data.

Proc component Payment risk processor, which is responsible for classifying the payment as routine/non-routine and making the selection of possible sets of authorization methods required for a specific non-routine payment. This processor communicates with other components of the payments operator system to obtain the information necessary to classify a

payment or calculate the risk of a non-routine payment (e.g., the history of previous transactions between the client and the seller, fraud detection systems). The *Proc* also uses the data for formal verification of payment feasibility (e.g., the balance of the account, the client consent to automate payments).

Auth component Authorization results analyzer that is responsible for storing patterns of correct authorization results for each individual user and for validation of the authorization done by the client. Instances of the *App* component running on client devices pass authorization data using defined API. The *Auth* component is used only for handling non-routine payments.

Trans component The payment processor responsible for carrying out the process of customer identification and authorization of a payment, which provides APIs for the seller *Sell* system. The *Trans* controls the PUSH communication used to transfer information necessary to carry out the entire process (i.e., payment authorization requests and payment status notifications) to the *App* component and to the seller *Sell* system.

4.3 Protocol for system communication

To prove the feasibility of the data flow related to transaction procedure proposed in Section 4.1, the complete communication protocol along with exchanged information and communication channels has been designed. The implemented communication protocol begins with a set of messages that initiate the process and then forks into two sequences: for routine and non-routine payments. The protocol is defined as follows:

1. An instance of the *Cam* component operating at a particular PoS sends a sequence of client facial images (*face*) registered at the PoS to the *Ident* component. Along with the *face*, *Ident* also receives information about the seller (*S*) and about the specific PoS.

$$Cam_{PS} \xrightarrow{(face, S, PoS)} Ident$$

2. *Ident* uses the database of biometric face templates to find the client ID *Kf*. Next, the *Ident* passes information to the *Trans* that a PoS of seller *S* should receive information about the client (*Kf*).

$$Ident \xrightarrow{(S, PoS, Kf)} Trans$$

3. *Trans* uses client ID provided by *Ident* (*Kf*) and the repository *ID* to find the client identifier attributed to her by the seller *S* (*Ks*). If the client did not register her identifier given to her by the seller *S*, the *ID* returns the payments operator's client identifier *K*.

$$Trans \xrightarrow{(S, Kf)} ID$$

$$ID \xrightarrow{(Ks|K)} Trans$$

4. *Trans* sends a notification to the *Sell* system of the seller *S* at the PoS that the client (*Ks* or *K*) has been identified.

$$Trans \xRightarrow{(Ks|K)} Sell_{S, PoS}$$

5. The *Sell* system of seller *S*, using an appropriate API, provides *Trans* with a debit request for client *Ks* | *K* and amount *X*. *Trans* confirms the request responding with a unique payment identifier #.

$$Sell_{S, PoS} \xrightarrow{(K|Ks, X)} Trans$$

$$Trans \xrightarrow{\#} Sell_{S, PoS}$$

6. *Trans* locks funds on the account of client *K* and requests payment classification from the *Proc* component.

$$Trans \xrightarrow{(S, PoS, K, X, \#)} Proc$$

Routine payment

7. *Proc* retrieves the necessary data from external systems of the payments operator, classifies the payment as routine, positively verifies formal conditions for the payment, and, in response, sends *Trans* information about the possibility to complete the payment in automatic mode (*auto*).

$$Proc \xrightarrow{(\#)=auto} Trans$$

8. *Trans* charges the customer's account and sends notification about the successful completion of the payment to the *Sell* system of seller *S* at the PoS and to all instances of the *App* component of the customer *K*. *Trans* pulls information about instances of the *App* component, which should be notified, from the *Devs*.

$$Trans \xrightarrow{(\#)=accepted} Sell_{S, PoS}$$

$$Trans(Devs) \xrightarrow{(S, PoS, X)=accepted} App_K$$

Non-routine payment

7. *Proc* retrieves the necessary data from external systems of the payments operator, classifies the payment as non-routine, positively verifies formal conditions for the payment, and in response sends information about the need to authorize transaction along with a list of possible sets of authentication methods (*auth*), to *Trans*. Algorithm for the calculation of sets of *auth* is described in details in Section 4.4.

$$Proc(Devs) \xrightarrow{(\#)=auth} Trans$$

8. *Trans* sends a notification about the need for payment authorization together with an indication of what authorization methods may be used on a particular device, to instances of the *App* component of the client *K*, indicated by *Proc* in the *auth* set.

$$Trans \xrightarrow{(S, PoS, X, \#)=auth} App_K$$

9. An instance of the *App* component used by the client *K* to input the authorization data sends the data (*D*) to the *Auth* component using appropriate API.

$$App_K \xrightarrow{(\#, D)} Auth$$

10. *Auth* verifies the correctness of the authorization data and sends information on the acceptance or rejection of the authorization to the *Trans*.

$Auth \xrightarrow{(\#)=accepted|rejected} Trans$

11. In the case of acceptance, *Trans* charges the customer's account and sends notification about the successful completion of the payment to the *Sell* system of seller *S* at the *PoS* and to all instances of the *App* component of the customer *K*.

In the case of rejection, *Trans* decides to recommence the evaluation of payment (step 7) or to terminate the payment processing with appropriate notification sent to the *Sell* system of seller *S* at the *PoS* and to all instances of the *App* component of the customer *K*.

Trans pulls information about instances of the *App* component which should be notified from the *Devs*.

$Trans \xrightarrow{(\#)=accepted|rejected} Sell_{S,Pos}$

$Trans(Devs) \xrightarrow{(S,Pos,X)=accepted|rejected} App_K$

4.4 Context-based selection of authorization methods

A task of *Proc* component is to classify payments into the classes of routine/non-routine and to construct sets of allowed authorization methods (AM) for non-routine payments, according to the current usage context (c.f. step 7 for non-routine payment in Section 4.3).

Two phases for the process of AMs sets building have been proposed. In the first phase, named *Types Selection*, based on trust, risk and convenience criteria, sets composed of the allowed authorization methods types (AMT) are calculated. In the second phase, named *Methods Selection*, sets composed of available AMs are generated based on the results of *Types Selection* and on the data received from *Devs* component regarding the availability of specific AMs on the client and seller devices. It is worthy to note that this phase results in sets of AMs, as opposed to single AMs, since in the case of a transaction with a higher level of risk, multi-factor authorization is required. More than one set of AMs can fulfill the given criteria; thus, finally, *Methods Selection* returns the list of sets of AMs.

AMTs analyzed during *Types Selection* have been organized in Fig. 3. The horizontal axis represents the required level of seller's trust to the client (denoted by *C*). The vertical axis represents the required level of client's trust to the seller (denoted by *S*). A given AMT can be selected if it is located in the chart at the point determined by the current *C* and *S* values, below that point, or to the left.

Fig. 3 AMT classification by the trust level to client and seller. Convenience ranking is denoted in square brackets

For example, if high levels of trust, both to client and seller are calculated, any AMT can be selected, including AUTO. AUTO represents the most convenient procedure, not requiring any action from the client. However, for such situation to take place, a number of prior successful transactions leveraging trust level must be conducted between the client and the seller. Thus initially, e.g., for low *S* and medium *C* levels, AMTs based on the knowledge attributes that are input on the client device (e.g., password, PIN), as well as all biometric AMTs used on the client device (fingerprint, voice), are available to the users.

The third criterion, not reflected in Fig. 3, is transaction risk level (*R*). It is assumed that low *R* implies single AM (AMT) choices, medium *R* implies choices of two-element sets, and high *R* implies choices of three-element sets (multi-factor authorization). Moreover, a convenience ranking of AMTs is denoted in Fig. 3 that is represented by values in square brackets (the lower value, the higher ranking position).

In the *Types Selection*, the following constraints for building AMT sets are used:

1. At least one AMT in the set has to fulfill the criteria of trust to the client and seller.
2. If more than one AMT fulfills the criteria of trust to the client and seller, the one that is higher in the convenience ranking is chosen.
3. The subsequent AMTs are chosen based on their positions in the convenience ranking, but:
 - (a) At most, one “knowledge” AMT is included in the resulting set.
 - (b) AMT “active” and AMT “passive” of the same type and at the same side cannot coexist in the resulting set.

- (c) AUTO method can be chosen only for $C = \text{high}$, $S = \text{high}$, $R = \text{low}$ (and obviously, it cannot appear in the set composed of more than one element).

According to the constraints described above, authorization rules engine generates all AMT sets that are compliant with C , S , and R security criteria (sets are composed of 1, 2, or 3 elements, depending on R). Then, AMT sets ranking is built, according to convenience criterion. To this end, convenience values of particular AMTs are summed up to become an aggregated convenience values for the AMTs sets.

In the *Methods Selection* phase, based on the calculated AMT sets, sets of allowed AMs are generated. To this end, information regarding the availability of specific AMs on the client and seller devices that are involved in the interaction, is received from *Devs* component repository and utilized. Since given AMT can appear only once in the set (and AMT “active” and AMT “passive” of the same type and at the same side cannot coexist in the AMT set), in the resulting set two AMs of the same type cannot be included. Therefore, strong authorization requirement is fulfilled (diversity of method types). The authorization rules engine returns optimal AM set, or ranking of AM sets, depending on the request.

Examples of parameters that impact criteria values and can be used in the approach described above, are listed in Table 2. Such values are calculated by *Trans* component. The criteria related to the situation context influencing the usefulness and quality of AMs have been used in the model for adaptable context-based biometric authentication dedicated for mobile devices and described in one of our previous articles [17].

Examples of AMs along with corresponding AMTs have been listed in Table 3.

Table 2 Examples of parameters impacting C , S , R criteria values

Criterion	Parameter
Level of trust to the seller	Global transaction number
	A number of transaction with a given client
Level of trust to the client	Regularity of transaction value (value vs. average, dispersion)
	Regularity of transaction location
	Regularity of transaction time
	Initial identification (RFID vs face)
	Behavioral characteristics (e.g., suspicious visual behavior)
Level of transaction risk	Transaction value
	Type of good (e.g., product vs. service, ease of resell)
Situation context influencing the usefulness and quality of given AM	Noise level
	Lighting
	Number of persons
User preferences	Public vs. private context

5 Proof-of-concept prototype

5.1 Technologies and prototype implementation

Verification of technical feasibility of the proposed architecture requires technologies that could be used to implement all the components of the system to be identified. The detailed description of these components is presented in Section 4.2. The overview of suitable technologies and details on the prototype implementation for each system component are presented below.

5.1.1 Seller-side components

Cam component requires image recording hardware and communication software capable of sending the image to the *Ident*. In the simplest case, it could use a regular RGB camera and software for network video streaming. Obviously, resolution and exposition parameters of the camera sensor should be adjusted to the conditions at a particular PoS. However, this solution requires a high-speed link (streaming full video frames) and implies the need to use this link in a continuous manner, even when there are no customers at the PoS.

A more efficient solution is to use software for face detection and face image extraction at the *Cam* side and to transfer to *Ident* face images only. It significantly reduces the required bandwidth and allows using it in a more efficient way. Nevertheless, the image processing algorithms require considerable computing power and are sensitive to imperfect lighting, which may significantly hinder the detection and extraction of the face from a 2D image. The most efficient algorithms employ a multipass approach, which repeatedly analyzes the same image and usually does not operate in real time, which in this application is unacceptable.

This problem can be solved using a modern camera with depth detection. The first widely available sensor of this type is Microsoft Kinect, and soon there will be also cameras based on the Intel RealSense technology. In both cases, the projector uses a structured infrared light and an additional infrared camera to gather information about the depth of individual parts of the image. The depth information simplifies the process of extraction of individual objects recorded by the camera. Complementing such depth-aware camera with software for human tracking, face detection and extraction, and for communication, provides a practically efficient implementation of the *Cam* component. In the prototype, the *Cam* component has been developed with a dedicated.NET software running on a mobile PC that uses the Kinect for Xbox One, as the depth-aware camera sensor.

Although the *Sell* component itself is not a part of the system, the proof-of-concept required a custom implementation of this component. Therefore, the prototype of the external sales/loyalty system of the seller has been developed as a native

Table 3 Examples of AMs with corresponding types

Side	AMT	Active/passive	Description
Client	Auto	Passive	Automatic; fully passive method
	Device	Passive	Passive verification of smartphone possession
	Device	Passive	Passive verification of wearable possession
	Device	Active	Tap-based verification of smartphone possession
	Biometrics	Active	Fingerprint collected with a smartphone
	Biometrics	Active	Voice recognition on the smartphone
	Knowledge	Active	Password typed on the smartphone
	Knowledge	Active	PIN typed on the smartphone
	Biometrics	Active	Behavioral biometrics
Seller	Biometrics	Passive	Face recognition
	Biometrics	Active	Fingerprint on the seller's device
	Knowledge	Active	PIN typed on the seller's device
	Biometrics	Active	Voice registered on the seller's device

application for the Android OS. The prototype implementation of the *Sell* component supports one seller-side authorization method (typing the client's PIN on the seller's device), and the ability to identify customers with an RFID tag.

5.1.2 Client-side component

In the prototype, the *App* component is a native application for the Android mobile OS. Instances of this component work on client's smartphones and require a permanent connection to the Internet via any wireless technology. The prototype implementation of the *App* component supports the following authorization methods: acceptance, PIN, fingerprint. Additionally, for the purpose of the experiment, the prototype implementation of the *App* component has been extended with features such as presentation of the account balance, the number of payments made, and diagnostic tools.

5.1.3 Operator-side components

The *Ident* component is server software running entirely on the infrastructure of the payments operator. For optimizing the number and size of images streamed by the *Cam* component simplified packet communication is used, instead of maintaining a continuous stream of data. A single package contains only a sequence of images of the face and is supplemented with additional data returned by the algorithms for detection and extraction of the face (e.g., the level of face detection “certainty”). The user identification algorithm on the low-level uses algorithms calculating the similarity of face images (Eigenfaces [16]) and their standard open source implementations (OpenCV [18]), and on the high level it employs a novel heuristic algorithm to identify user from a series of low-level recognitions (possibly ambiguous) streamed at

real time. In the prototype, the *Ident* component is implemented in .NET technology.

All the other operator-side components have been implemented on Java platform and operate within a Restlet server. *Auth*, *ID*, and *Devs* components are RDBMS software. Optionally, these components provide an API to allow remote modules, such as instances of the *App* component, to manage the data. *Trans* component uses PUSH communication based on external communication service, namely Google Cloud Messaging. Communication requirements of other systems of the payments operator used by the *Trans* component (e.g., a fraud detection system) should be taken into account as well. However, they have not been included in presented implementation, but instead, *Trans* component implements functions of an electronic wallet and client (and seller) transaction history analyzer. This way, the prototype simulates the use of external systems of the payments operator which can be used for further experiments involving the functionality currently not shared by these systems.

For implementation of *Proc* component, if the goal is to use authentication method selection logic as described in Section 4.4 (or other logic with similar level of complexity), and at the same time, frequent changes in authorization method selection rules are not expected, then any of the major imperative languages and corresponding platform can be used. However, if the logic of authorization method selection rules will evolve during the operation of the system and the number of rules will grow, it may be reasonable to represent the rules using a dedicated language and use a declarative rules engine library (e.g., Drools or JRules). Such an engine would become the key element of the *Proc* component and the ease of its integration with the rest of the code of the component may be an important technical aspect. In the prototype, the *Proc* component has been attached to simulated “external systems” of the payments operator, that has provided values of the *C*, *S*, and

R parameters required by the *Proc*. To mimic real fraud detection and client profiling systems, the values have been calculated taking into account data such as total number of transactions of the client and the seller, number of transactions between the client and the seller, typical amount of transaction between the client and the seller, and transaction amount.

5.1.4 Communication

Communication between all the Java-based components on the payments operator side has been implemented using internal API—based on Java interfaces and methods (solid arrows in Fig. 2). Public interfaces have been implemented as a REST API (dashed arrows in Fig. 2). Push notifications sent to *App* and *Sell* components have been implemented using Google Cloud Messaging service (double-line arrows in Fig. 2). *Cam* and *Ident* components communicate with each other using methods provided by WCF Web Services (dashed-and-dotted arrows in Fig. 2).

5.2 Prototype deployment

The system has been deployed at a real location (a local bar located at a university campus) and used by 22 users recruited for the experiment from existing and new clients of the bar, for the period of 16 days. The participants were interviewed at the beginning and at the end of the experiment. During the experiment, they have performed 251 payments, which is more than 10 payments per participant. Therefore every participant was able to provide a valuable feedback about the system. Overall, the gathered feedback confirms the feasibility of the proposed concept and its practical usefulness. The passive client identification based on face images proved to be working properly, but the participants noted that it could be faster—to become invisible to the user, the passive identification would need to be done in under 5 s time, which was rarely achieved with the prototype. Initial results of the ongoing quantitative evaluation show that for routine payments the time P2 (payment) was near instantaneous, but the time I2 (identification) was usually above 20 s. Therefore, the next phase of this research is planned to test more efficient face recognition algorithms and to use of additional parameters that could be passively measured or calculated from face images and human tracking stream.

5.3 Lessons learned

The main goal of the reported qualitative evaluation is to verify the technical feasibility and the business viability of the proposed system. The proof-of-concept deployment and evaluation have confirmed the technical feasibility, but what is even more important at this stage, they have confirmed that the proposed system along with the transaction procedure

introduces added value for all actors of the process, which proves its business and practical potential.

5.3.1 Payments operator

With the introduction of the described concepts, the payments operator gains a tool for dynamic assessment of the type of payment (routine/non-routine) that goes far beyond the model adopted for contactless payment cards (hard limit quota). By dynamically assessing the level of risk of a payment, the tool can be used more widely than just to make a decision regarding payment classification as routine or not. The risk of a payment and trustfulness of the participants allow to dynamically select the most appropriate authorization methods of the payment.

Expanding the system with a mechanism for dynamic selection of authorization methods for non-routine payments significantly increases the value of the overall solution for the payments operator. Due to the dynamic assessment of trustfulness and risk, it is possible to build rules for dynamic selection of authorization methods, which take into account various methods supported by client's devices, as well as methods supported by other devices in the client environment (e.g., passive face recognition at a seller's location). It is possible to use multiple authorization methods simultaneously, which, if necessary, allows for greater authorization reliability. With the ability to dynamically compose sets of payment authorization methods, depending on the riskiness of a specific payment, it is possible to maintain a controlled balance between security and convenience of payment for the client and the seller.

5.3.2 Seller

From the point of view of the seller, the automation of routine payments does not change much. The vast majority of time devoted by the seller to customers is included in the process of handling orders, not on the payment itself. Nevertheless, the fact that the customer has to be identified for the purpose of payment can be used to introduce improvements in the customer service process. To automate the routine payments, the identification of the customer must be at least just prior to initiating the payment by the seller. But by moving the moment of identification to the beginning of the service process and by providing the seller's sales/loyalty system with information about customer identification, our system makes it possible to use a wide range of tools to support personalized service—starting with the standard loyalty tools (e.g., “You have bought from us 5 coffees this week, so this time you can have a cake at half price”) and ending with full personalization support (e.g., “Good morning, Mr. White. Do you want to order the usual?”) regardless of the knowledge and memory of individual employees serving clients.

5.3.3 Customer

It is the end customer for whom the presented system introduces the highest value. She gains on every functional element of the system, starting with fast and convenient routine payments and the ability to use convenient authorization methods for non-routine payments, to full personalization of service and automation of loyalty procedures.

It has to be noted that regardless of particular system, general security and privacy concerns impacting the social acceptance of biometric identification have to be analyzed before mass deployment in retail trade. Conditions required by individuals to use biometric methods at PoS have to be analyzed, and the selection of authorization methods seen as a reliable and privacy-preserving in specific circumstances has to be made.

6 Conclusions

In this paper, technical feasibility of a system for context-aware passive payment authorization in brick-and-mortar retail stores has been presented. Aiming to surpass the existing contactless payment solutions, three main requirements have been identified, namely automation of routine payments, passive customer identification with ambient face recognition, and intelligent control of payment security level. The data flow for a new customer service procedure has been formalized, in which the customer is identified before making an order and the payment authorization is shorter or more convenient than in traditional solutions. A complete architecture of a system that is able to implement such transaction procedure has been designed as well. The system design has been detailed with a thorough specification of a communication protocol and a method for context-based selection of authorization methods (including detection of routine payments that can be fully automated). The feasibility of the proposed solution has been verified by implementing a proof-of-concept prototype according to the requirements and the design and deploying it in a real world environment.

The lessons learned during the development of the prototype and the feedback gathered from users of the prototype prove that the system is feasible and offers the following key advantages:

- Makes it possible to maintain a controlled balance between security and convenience of payments
- Enables personalized customer service
- Makes the payment process faster and more convenient for the client
- Provides all actors of the process with added value which proves its business and practical potential

As a next research step, quantitative evaluation of system technical parameters and security issues is planned, as an extension of the qualitative evaluation presented in this paper.

Acknowledgements The research was supported by Santander Universidades and Bank Zachodni WBK. We thank our colleagues from Bank Zachodni WBK and Symetria who provided insight and expertise that greatly assisted the research. We also thank Rafał Wojciechowski from our Department for support with implementation and deployment.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. Alliance, Smart Card (2012) EMV and NFC: Complementary Technologies that Deliver Secure Payments and Value-Added Functionality. A Smart Card Alliance Payments Council White Paper
2. Apple (2014) Apple Pay, <http://www.apple.com/apple-pay/> Accessed 09–06-2016
3. Samsung (2015) Samsung Pay, <http://www.samsung.com/us/samsung-pay/> Accessed 09–06-2016
4. Google (2015) Android Pay, <https://www.android.com/pay/> Accessed 09–06-2016
5. EMVCo LLC (2014) EMV Payment Tokenisation Specification—Technical Framework, version 1.0, Mar 2014
6. Google (2015) Google Hands Free, <https://get.google.com/handsfree/> Accessed 09–06-2016
7. Christopher Smowton, Jacob R. Lorch, David Molnar, Stefan Saroiu, and Alec Wolman (2014) Zero-effort payments: design, deployment, and lessons. In: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14). ACM, pp. 763–774
8. Anh Tran, Quoc (2016) Finnish Grocery Retailing Market Assessment for the Deployment of Payment Innovation: Case: Uniqul Face Recognition Payment Application., Lahti University of Applied Sciences
9. Bowyer KW (2015) Selfies to emerge as both payment, anti-fraud solution, Biometric Research, Virtual Special Issue, Elsevier
10. Lucnova (2015) FreshX, <https://www.freshxapp.com/> Accessed 09–06-2016
11. Liquid Pay (2015) Liquid Pay, <http://www.liquidpay.com/> Accessed 09–06-2016
12. Fujitsu (2011) PalmSecure, <http://www.fujitsu.com/us/solutions/business-technology/security/palmsecure/> Accessed 09–06-2016
13. Biyo (2014), Biyo Wallet, <http://biyowallet.com/> Accessed 09–06-2016
14. Lee J. (2015) JCB piloting Fujitsu palm vein authentication technology for payments, Biometric Update, Biometrics Research Group, <http://www.biometricupdate.com/201510/jcb-piloting-fujitsu-palm-vein-authentication-technology-for-payments> Accessed 09–06-2016
15. Bolle RM et al. (2013) Guide to biometrics. Springer Science & Business Media
16. Turk MA, Pentland AP (1991) Face recognition using eigenfaces, Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91, IEEE Computer Society Conference on, 586–591
17. Wójtowicz A, Joachimiak K (2016) Model for adaptable context-based biometric authentication for mobile devices. Personal and Ubiquitous Computing, 20(2), Springer, 195–207
18. OpenCV 2.4.13.2 documentation (2016) Face Recognition with OpenCV, http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#eigenfaces Accessed 09–06-2016