

Transaction Authenticator Pattern

Fabricio A. Braz¹, Eduardo B. Fernandez², Diogo C. Rispoli¹

¹Faculdade Gama - Universidade de Brasília (UnB)
Área Especial de Indústria - Projeção A - UnB - Setor Leste do (Gama) - 72444-240 -
Brasília - DF - Brasil

²Department of Computer & Electrical Engineering and Computer Science
Florida Atlantic University
777 Glades Road - Boca Raton, FL 33431, USA

fabraz@unb.br, fernande@fau.edu, drispoli@gmail.com

Abstract.

Every internet intensive system which deals with valuable transactions is an attack target. Several attacks focused on such transactions have been widely reported, most of them supported by stolen credentials. Fortunately, some of these systems are empowered by an additional control layer that requests an extra authentication to finish the transaction execution process. In this paper we present the Transaction Authentication pattern that provides a solution for organizations that face the same problem in the context of valuable transactions.

1. Intent

Apply an extra authentication step to those transactions considered sensitive because of their privacy requirements or monetary value.

2. Example

The Acme Bank Ltd. provides to its customers a web banking service to make many kinds of transactions varying from a simple account statement to an interbank fund transfer. Obviously, such service is one of the attacker preferred targets, since she can perform a spurious interbank fund transfer transaction, withdrawing the money in the destination account bank (usually created with a fake identity) shortly after the transfer, turning the finance recoverability near zero. Frauds like that may imply severe risk to business sustainability. What can the bank do to stop these attacks?

3. Context

Financial systems or other systems which deal with valuable information and where users perform remote transactions.

4. Problem

Credential stealing to perform illegal transactions is a serious problem in financial institutions. Combining actions such as malware and social engineering has been a very effective approach to collect user credentials [Ver13, Moh12], but avoiding such actions

poses a very complex and unsolved problem [Moh12]. How to prevent an attacker from executing valuable transactions with stolen user credentials? Information systems may perform transactions which deal with sensitive data, and are very valuable from a privacy view point or because of their financial value. After a successful attack, it can be almost impossible for the organization to recover from the prestige damage or the financial loss.

4.1. Forces

The following forces apply to the possible solution:

- *Flexibility*: a system may deal with transactions with different values, and one size does not fit all . We need to offer a solution that applies an appropriate transaction authentication approach, depending on the transaction's value.
- *Doubt*. The attacker may using valid stolen credentials and we should not accept dubious credentials as valid.
- *Transparency*. The users do not know in advance what type of authentication will be applied.

5. Solution

Use an additional step of authentication which only applies to valuable transactions. Once such transaction is requested, its execution would be committed only if the subject successfully proves he is who he declared to be. Such authentication should employ a different approach (i.e. SMS, Token, Biometrics) from the one already used to create the subject interaction session [Sch06].

5.1. Structure

Figure 1 shows the class diagram for this pattern. The interface *ITransaction* defines the basic transaction operations. Every transaction must implement that interface. The abstract class *ValuableTransaction* implements *ITransaction* and it also implements the *execute* operation. In the body of this method the operation *authenticateTransaction*, from *TransactionAuthentication* class, is called, and, depending on method call result given by *validateUser* operation, the system calls *perform*, for successful authentication, otherwise it invokes *reject*.

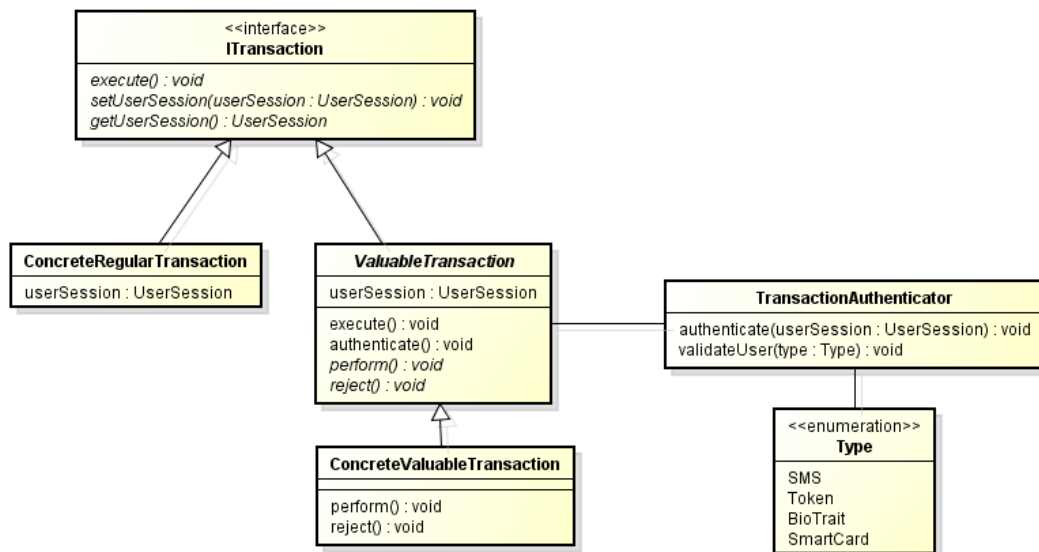


Figure 1 - Transaction Authenticator Class Diagram.

The *perform* operation is responsible for carrying out all the steps to accomplish the valuable transaction (i.e. to request the business layer to process the transaction). The duty of *reject* is to assure that the transaction request will not be performed, including the resources deallocation, event logging, and so.

You might note that both methods must be implemented in the concrete class, such as *ConcreteValuableTransaction*. It means that the concrete class will have the same *execute* method as its superclass, but it has full control over the transaction sequence.

5.2. Dynamics

Figure 2 presents a sequence diagram which describes the dynamics of Transaction Authenticator pattern.

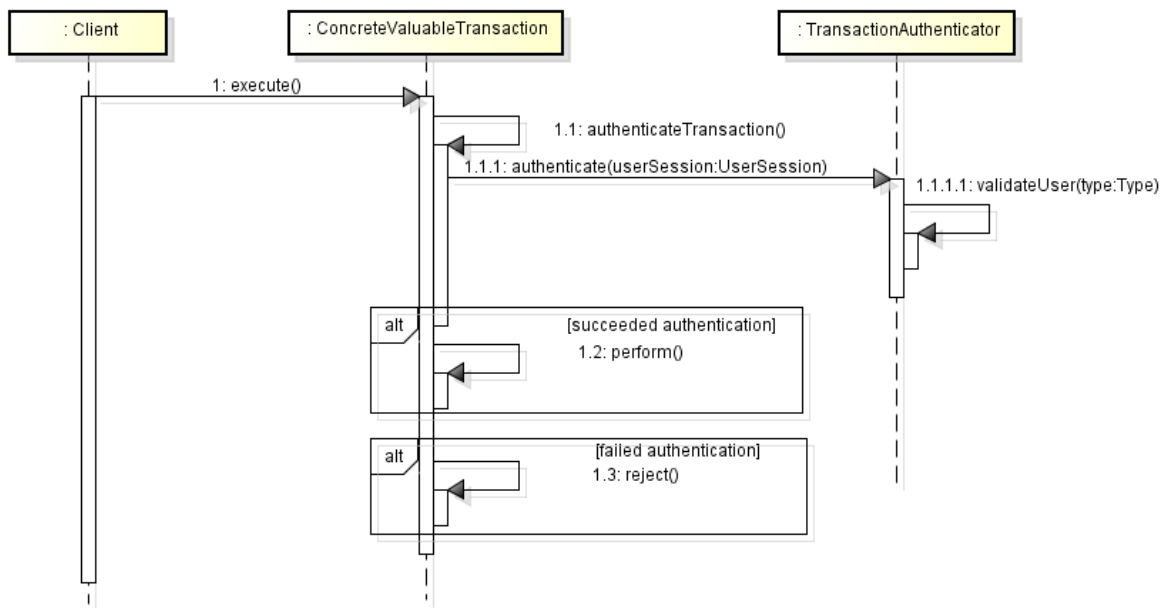


Figure 2 - Transaction Authenticator Sequence Diagram for the use case Perform Transaction.

6. Implementation

The pattern TransactionAuthenticator has been implemented in Java. The Figure 3 shows an abstract class called ValuableTransaction.java. The body of the concrete method *execute* enforces another authentication layer, whose result will impact in the transaction execution progress, carried by perform and forsake.

A complete version of the sample project can be downloaded at <https://code.google.com/p/transaction-authenticator>.

```

package br.com.authentication.transaction.impl;

import br.com.authentication.authenticator.TransactionAuthenticator;

public abstract class ValuableTransaction implements ITransaction {

    private UserSession userSession;

    protected abstract void perform() throws TransactionException;

    protected abstract void reject();

    public void execute() throws AuthenticatorException {
        try {
            authenticateTransaction();
            perform();
        } catch (TransactionException e) {
            reject();
            throw e;
        }
    }

    private void authenticateTransaction() throws AuthenticationException {
        TransactionAuthenticator.getInstance().authenticate(getUserSession());
    }

    public UserSession getUserSession() {
        return userSession;
    }

    public void setUserSession(UserSession userSession) {
        this.userSession = userSession;
    }
}

```

Figure 3 - ValuableTransaction.java

7. Known Uses

Brazil is one of the biggest sources of banking Trojans, motivated by millions of online banking customers and an ineffective legislation which turns it difficult to punish attackers [Bes13]. As a consequence, banks have been losing significant amounts of money driven by stolen credentials.


In order to reduce the financial damage, the banks are compelled to put in operation many security controls. One of those controls is a transaction authentication through a secret SMS. Such control can be found in two banks: Santander [San13], and Itaú.

Santander's web banking workflow follows the steps described below:

1. The user supplies the transaction details, such as the destination account, the monetary value, and user credentials to be verified for fund transfer.
2. The system checks the type of transaction.
 - 2.1. In case of a valuable transaction, the system sends to the user's mobile phone number a SMS with secret. That secret must be supplied by user to the web banking form, presented in Figure 4, to conclude the transaction.

2.2. In case of a regular transaction, the system employs regular business rules to check regular needs, like an enough balance, and concludes the transaction.

SMS Token

Insira abaixo o Código do SMS Token exclusivo para esta transação que foi enviado para o celular: 61 XXXX - 7899 

Caso não receba o SMS Token:

- » Verifique se o número de celular esta correto;
- » O SMS poderá levar até 1 minuto para chegar (de acordo com a disponibilidade de sua operadora);
- » O código do SMS Token para esta transação é válido por 4 horas;

Em caso de dúvidas, [clique aqui](#) ou entre em contato com a Central de Atendimento Santander:
4004-3535 (Capitais e Regiões Metropolitanas)
0800 702 3535 (Demais Localidades)

Digite o código do SMS Token:

[voltar](#) [limpar](#) [enviar](#)

Figure 4 - Santander Web Banking.

In addition to Santander, we identified three financial institutions that also implement transaction authenticator. The first is a small bank from Brazil Federal District called Banco Regional de Brasília [Brb13]. Its web banking features a similar workflow to the one from Santander. The same can be seen in other web banking systems from financial institutions much larger than the latter, such as BBVA [Bbv13] and DBS [Dbs13], which suggests the transaction authenticator pattern would fit to any business size.

It is important to state the examples go beyond the financial field. It can be seen in the frequent flyer services, like the one provided by TAM [Tam13], whose redeem web site, also implies a transaction authenticator through SMS, in order to commit a transaction.

9. Consequences

Flexibility—we can apply authentication only to some transactions that we consider very valuable.

Doubt—by applying further authentication we can make sure that the transaction requester is legitimate

Transparency—The users do not know in advance if their transactions will be authenticated again and an attacker cannot be prepared.

10. Related Patterns

Authenticator [Fer13]. When a user or system (subject) identifies itself to the system, how do we verify that the subject intending to access the system is who it says it is?

Spoofing web services (misuse pattern) [Mun11]. A web service spoofing misuse tries to impersonate the identity of a user by stealing his credentials, and then makes requests in his name with these credentials, with the intention of accessing a victim's web service.

Acknowledgments

We thank our shepherd, Antonio Terceiro, for his valuable comments that improved our paper.

References

- [Aba12] ABA Banking Journal. Available from: <http://www.ababj.com/tech-topics-plus/adptive-authentication-product-protects-against-account-takeovers-3499.html>
- [Bbv13] Banco Bilbao Vizcaya Argentaria, S.A (2013) “Microsite SMS Token”. Retrieved July 1st, 2013 from: <https://www.bbva.pt/TLEU/tleu/jsp/pt/porN/bbvanet/smstoken.jsp>.
- [Bes13] Dmitry Bestuzhev. (2009) “Brazil: a country rich in banking Trojans”. Retrieved June 27, 2013 from: http://www.securelist.com/en/analysis/204792084/Brazil_a_country_rich_in_banking_Trojans.
- [Brb13] Branco de Brasília (2013). “BRB- Banco de Brasilia”. Retrieved July 1, 2013, from <http://www.brb.com.br>
- [Dbs13] DBS Bank Ltd (2013) “DBS iBanking Message FAQs | DBS Singapore”. Retrieved June 28, 2013. From <http://www.dbs.com.sg/personal/ibanking/additionalinfo/faq/ibmessage/default.page>
- [Fer13] E. B. Fernandez, *Security patterns in practice - Designing secure architectures using software patterns*. Wiley Series on Software Design Patterns, June 2013.
- [Moh12] J.G. Mohebzada, A. El Zarka, and A.H.Bhojani, A. Darwish, (2012), "Phishing in a university community: Two large scale phishing experiments," *Innovations in Information Technology (IIT), International Conference on* , vol.?, no., pp.249,254, 18-20 March 2012
- [Mun11] Jaime Muñoz-Arteaga, E.B. Fernandez, and Hector Caudel, "Misuse pattern: Spoofing web services", *Asian PLoP 2011*.
- [San13] Banco Santander S.A. “Banco Santander Brasil – Banco do Juntos”. Retrieved June 29, 2013, from <http://www.santander.com.br>.
- [Sch06] Schumacher, M., Fernandez, E. B., Hybertson D., Buschmann F., and P. Sommerlad. (2006), “Security Patterns: Integrating Security and Systems Engineering”, Wiley.
- [Tam13] TAM Linhas Aéreas S/A. “Tam Linhas Aéreas – Paixão por Voar e Servir”. Retrieved June 27, 2013. From <http://www.tam.com.br>.
- [Ver13] Verizon. (2013) “2013 Data Breach Investigations Report”, Online in: <http://www.verizonenterprise.com/DBIR/2013/>