WHITE PAPER

Advanced Security Design for Financial Applications

Abstract

Providing foolproof security for financial applications is a rigorous activity. Security architect needs to consider various design considerations to make the applications bullet proof. The white paper provides comprehensives coverage of various security aspects of financial applications. We provide various security design considerations, best practices, techniques and discuss a case study. Financial applications referred in this white paper include web applications, financial portals and other finance domain related online applications.

Infosys
be more

Advanced security design considerationsfor financial applications

Let us look at some of the advanced security design considerations while designing financial applications.

Two factor authentication

Two factor authentication is needed to step up the primary authentication with a supplementary authentication for financially critical operations. In some cases regulatory body make it mandatory for two factor authentication for few transactions like fund transfer. The two factor authentication can be implemented with any mechanism which is fool-proof. Some of the two factor authentication mechanism include.

- One Time Password (OTP) – When the transaction is initiated, the application calls the OTP server through service call of ESB, and OTP server connects with the SMS gateway to generate the OTP to the customer'sdevice. Once the customer enters the OTP, then it is again validated by application through service call to OTP server via ESB. Once authentication is successful, the transaction is allowed to proceed.
- Security questions – When the transaction is initiated, the application prompts few security questions, which the user has already set up. Once the security answers are validated, the transaction is allowed to proceed.
- Hard tokens – When the transaction is initiated, the application prompts for token number generated, on the hard token which the user holds. The token is authenticated, and then the user is allowed to proceed the transaction.

Digital Certificate based security

Certificate based security is a security mechanism wherein we leverage the digital signature certificate(DSC) held by the user for authenticating the user for secure transaction. Ideally the user will have to hold the digital signature certificate issued by Certifying Authorities in the form of hard token like pen drive. When the user wants to do the transaction, he would plugin the hard token into the physical device like desktop/laptop, and the web applicationshould get the digital signature certificate from the hard token, and prompt the user to sign the digital certificate for the transaction.

Let us take an example of a user who wants to do fund transfer transaction with DSC.

As a foremost step, the user would need to get the DSC from one of the Certifying Authorities, approved by the Office of Controller of Certifying Authorities (CCA). Then the user needs to register the certificate with the finance application by inserting the hard token into the device. The application would get the private key of the certificate from the hard token, and gather the customer details of the logged-in user from its internal application, and prompt the user to register the certificate. Once the user the confirms the registration, the customer details can be encrypted with public key provided by the user, and sent to another server application (DSC Manager) through service calls, which can manage the binding of customer and digital certificates, and authenticating whether the user has provided a valid certificate. After the DSC registration is successful, the user can do transactions based on digital signature certificates. The user logs into the portal, and assume he wants to do a fund transfer transaction. He would insert the hard token, and then fills all the details for the fund transfer on the portal. When he submits the transaction, the application would read the DSC from the hard token, and prompt him to sign the transaction details.

The below data can be prompted to the user,

Content to be signed:

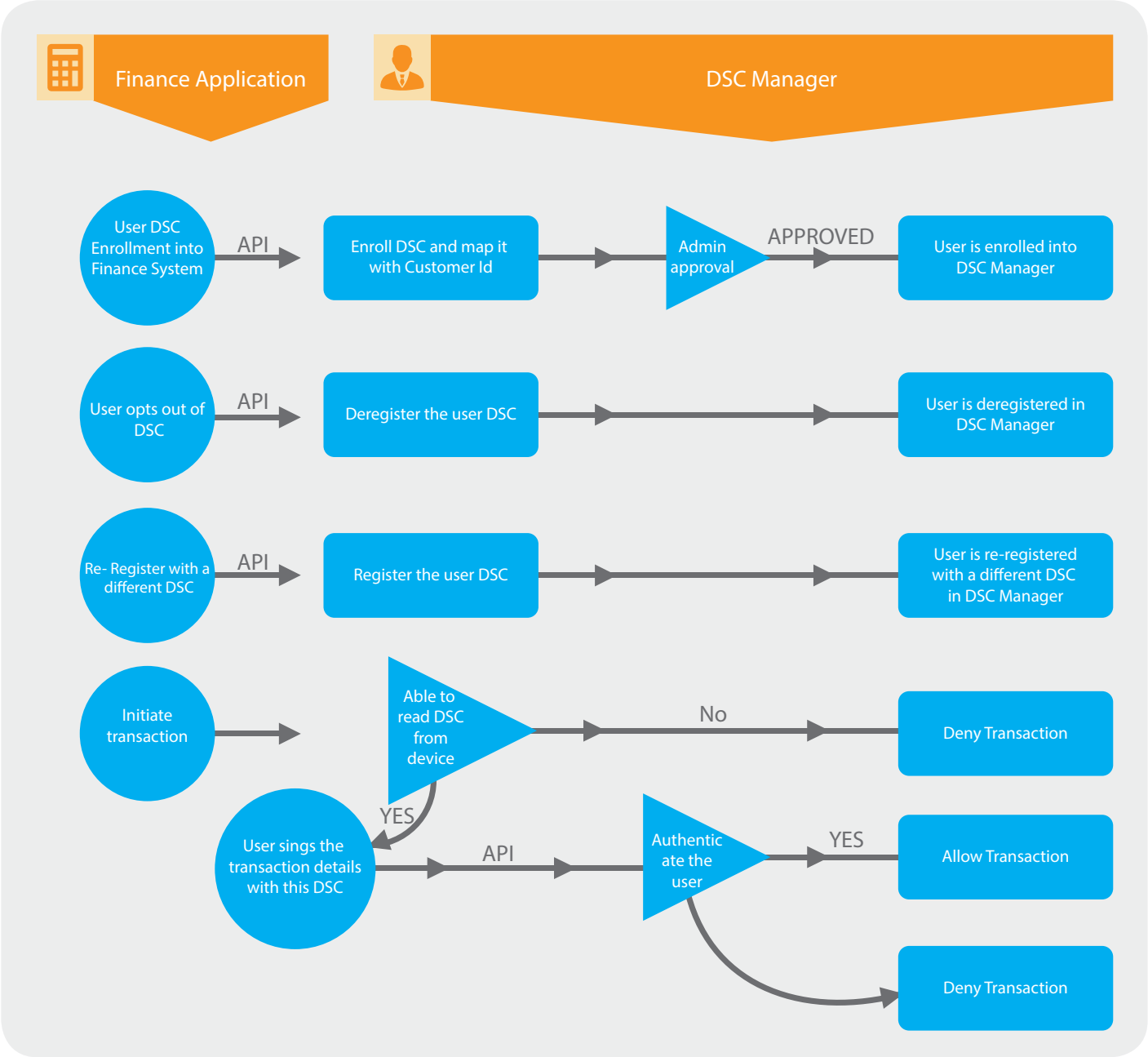
From: XYZ; To: ABC; Amount: Rs. 20000 ;Mode: NEFT;Date:09-Oct-2015;Transaction Effective Date: 09-Oct-2015

Select the certificate below and sign the transaction.

Name: XYZ Issuer: ABC Authority Serial No.: 23455

When the user signs the transaction with the certificate, the application need to prompt for the password to access the private key, using which the data above will be encrypted.

A sample flow for this is shown below:



Benefits of DSC Security:

Feature	Benefits
DSC Enrollment	The user enrolls the digital signature certificate in banking system, which will be mapped with his unique customer id. The user is confident of the security system that the transaction can happen only on providing the DSC which he holds.
Authentication mechanism	Gains the confidence of both bank and customer from authentication perspective, and the bank can also be confident that the transaction happened only within the knowledge of customer, as no one else is supposed to have his own digital signature certificate.

Adaptive authentication (risk evaluation and multi factor authentication):

Adaptive authentication is a comprehensive authentication mechanism to analyze the risk associated with a transaction based on the device print, geolocation, user's past behavior etc. An adaptive authentication (AA) system could be developed which would calculate the risk score, based on the risk parameters. Based on the risk score, the bank can step up the authentication if necessary. The policies can be set in adaptive authentication framework, which can precisely identify what kind of authentication mechanism will best suit based on the risk profile of the customer.

Adaptive Authentication system wouldmeasure authentication related risks and determines level of authentication required based on risk, policies, and customer segmentation. You may ask a question, how the adaptive authentication mechanism could know the risk associated with a transaction. Ideally this AA system should keep on capturing the user behaviour and device prints, which would enable to detect any unusual transaction.

Assume a user has done the transaction only from India for the past two years, and suddenly a transaction is detected from another country say Namibia. Then adaptive authentication system should consider this as a suspicious transaction. But the customer may have indeed travelled to Namibia also, and carrying the transaction. So AA system, would prompt for additional authentication checks like One Time Password, Challenge Questions, or any other step up authentication system. What authenticate an AA system would trigger depends on the policies which we configure in the AA system. The user would be authenticated for step up authentication. Once authenticated, the user can continue the transaction.

AA system captures device finger print like browser info, display settings, time-zone settings, installed software, regional language settings, IP address, cookies etc. For a mobile device, it ccaptures mobile finger prints like SIM ID, hardware ID, Wifi Mac Address, Address book size etc, for risk analysis of mobile transactions. AA system captures the user behaviour and tracks the transactions and its parameters, which could help to challenge the user for any unusual behavior or transaction, when the user tries to initiate the transaction.

AA system can be further enhanced by centralizing the list of IP address, device prints etc., from where fraudulent transactions have been detected consistently. If AA System should encounter a transaction from such IP address or device, then the transaction can be denied straightaway, rather than prompting to step-up the authentication.

When the customer is registered on the finance application, the user would also need to be registered into the AA system simultaneously. The user can be registered using API call of AA system, which can be called from the finance application.

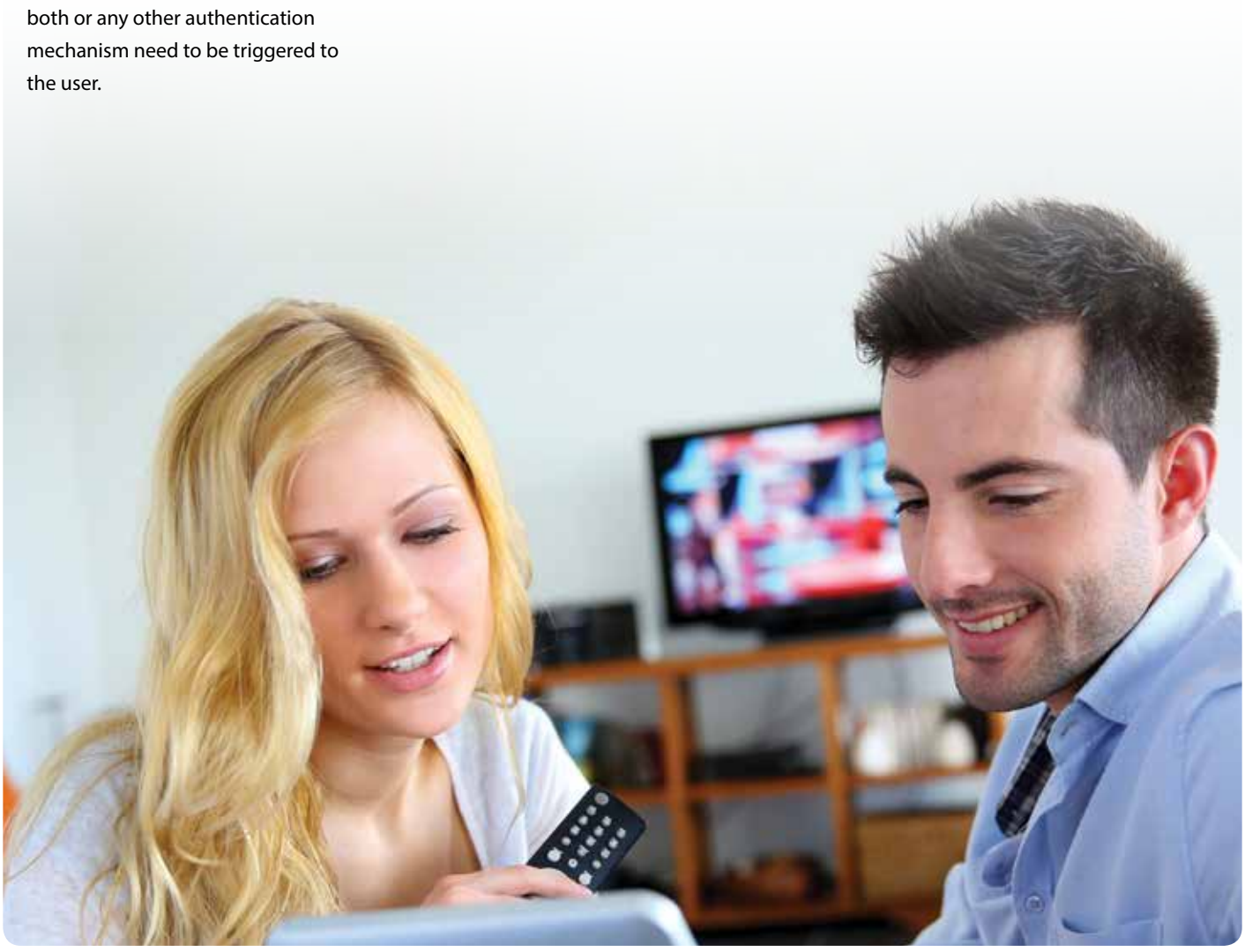
Once the user is registered in the AA system, it is time to capture the user's profile information also, and pass them to the AA system through API calls. What kind of profile information would be required by an AA system – Well, it depends on what kind of step authentication are required, and what profile information is required to execute it.

Consider a case, wherein the finance institution wants to implement two authentication mechanism as part of adaptive authentication - challenge questions (CQ) and One Time Password (OTP) authentication. For CQ authentication, what profile information are needed? The challenge question and answer set by the user would be the profile information, which is required by the AA system from the finance application. Similarly to implement OTP authentication, what profile information would be required? Yes, it is obviously the mobile number. If OTP need to be triggered through Email also, finance application need to send both mobile number and email id of the user to the AA system.

Now let us see, how to implement CQ in a finance application for adaptive authentication. The list of challenge questions would be configured in the AA system. You can separate the list of related questions to a group, and multiple groups can be configured in the AA system. Ideally the AA system should have an interface to add groups, and add or edit the questions within the group. The finance application can fetch all these list of challenge questions through an API call, and display the challenge questions to the user. The user can be mandated to answer at least one question in each of the group, which can be passed to the AA system through an API call. Now you may ask a question what would happen if the user forgets the challenge answers which he had already set. Portal can provide an option to the customer to reset the CQ with a mandatory OTP.

For implementation of OTP, as one of the means of adaptive authentication, the mobile number or email can be passed to the AA system, either during registration process of the user to the AA system, or by prompting the user to set up the mandatory profile information, which can be passed to the AA system.After getting all the user information, AA system should have the following capabilities. For each of these capabilities, the AA system should expose a service to the finance application.

- AA system should be capable of calculating the risk score of the user based on the risk parameters, and return the corresponding adaptive authentication best suited to customer profile and risk. For e.g., the policies would determine whether challenge question, One Time Password, or both or any other authentication mechanism need to be triggered to the user.
- Once the authentication mechanism is decided, the AA system should be capable of gathering all details required to trigger the corresponding authentication mechanism. For e.g.,if the authentication mechanism triggered by AA system is Challenge Questions, then a service should be exposed from AA system, so that the finance application can invoke the service, and fetch the challenge questions to be answered by the user.
- Once the user inputs the data required by the adaptive authentication system, the AA system should be capable of getting the data and authenticate against the corresponding authentication mechanism under it purview.
- If the authentication mechanism is outside the purview of the AA system, then the authentication result should be capable of capturing the notification of authentication result from the web application, which would help in capturing the user behavior, and this could again help in calculating the risk score based on user behavior. For e.g., the financial institution can implement OTP as separate entity, rather than binding it to the AA system – the simple reason for this principle, could be that the OTP server could be used by different systems for any mandatory OTP required by the finance institution, rather than an adaptive authentication. In this case, if OTP verification is successful/failed, the result need to be notified to the AA system by the finance application.



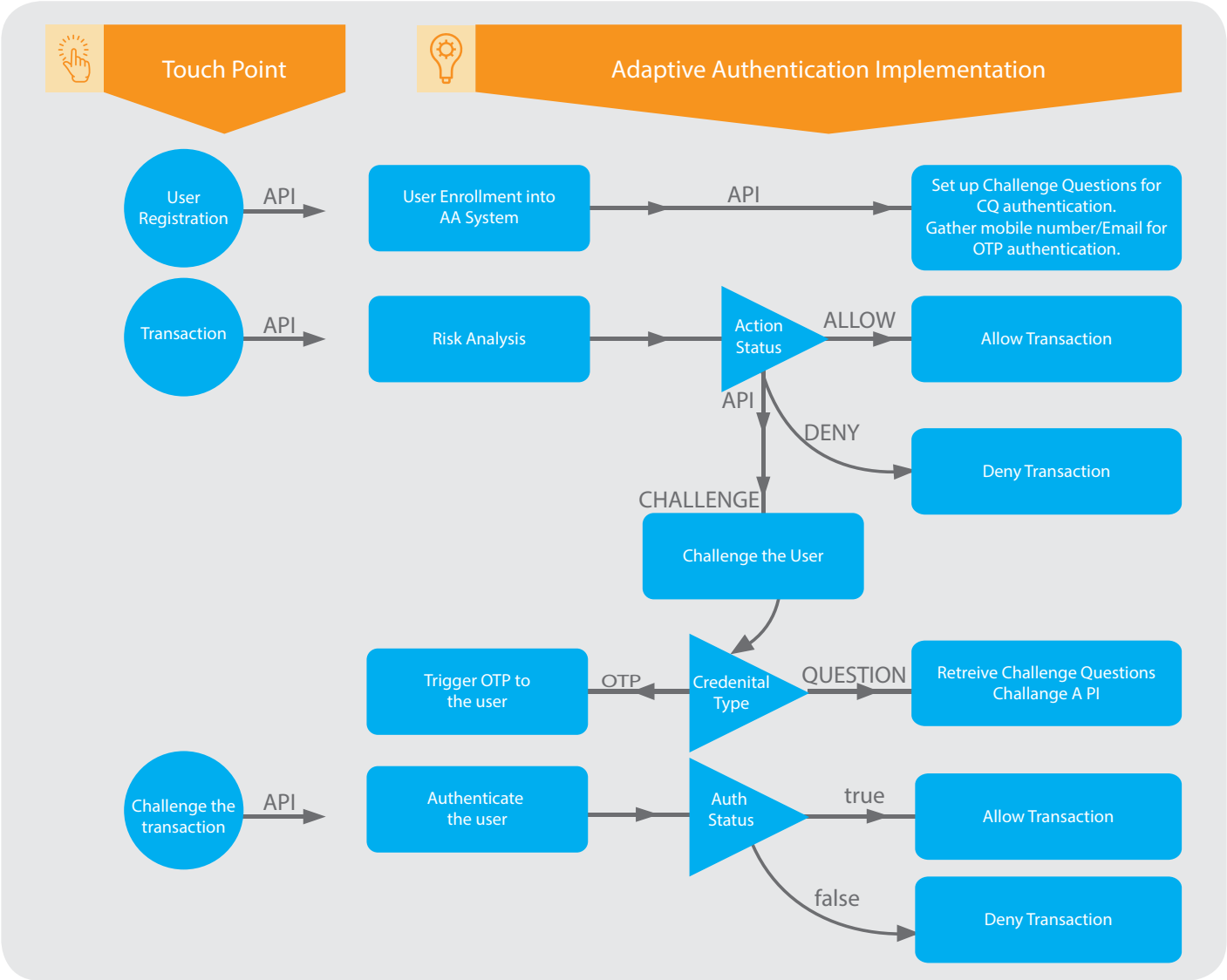
If the AA system is designed and implemented with all the above things in mind, the AA system is now ready to evaluate the risk score, suggest suitable adaptive authentication required, and authenticate the user based on it.

We would consider two cases of adaptive authentication – Say OTP, and Challenge Questions.

If AA system considers the OTP would be sufficient for adaptive authentication, then AA system can itself trigger an OTP to the user, if OTP implementation is bound within the adaptive authentication. Once the user enters the OTP on the portal, it will be authenticated with the AA system. If the finance institution already have an enterprise OTP system, and would like to go through it only, and then AA system's OTP mechanism can be skipped.

In that case, AA system would respond back to portal asking to trigger the OTP, and inform back the OTP authentication success/failure status through notify API call. If AA system considers that challenge questions would be the most opt adaptive authentication, then AA system would inform back to portal to trigger CQ (Challenge Questions). The portal can get few questions, which the user has already set through the portal. The user would now answer the questions, and gets authenticated from AA system.If AA system considers that the transaction as very high risk, then it can take a decision to prompt for both CQ and OTP, based on the policies. In that case, both security mechanism need to be authenticated before allowing the transaction.AA system can consider a transaction as safe, and in this case it will respond back with 'ALLOW' status to the portal. Now, it is the portal's turn to decide whether step up authentication is required or not. You may now ask a question that if AA system considers a transaction as safe, why would portal need to again bother about step up authentication. Regulatory body under which the bank operates may ask for a mandatory step up authentication for few critical transactions like fund transfer. In that case, portal have to prompt for a step up authentication which it considers as most appropriate. Most banks go with a mandatory OTP in case of compliance requirement from the regulatory body.

A sample AA flow is given below. Adaptive authentication of OTP and Challenge Questions mechanism is considered in this workflow



Benefits of AA system

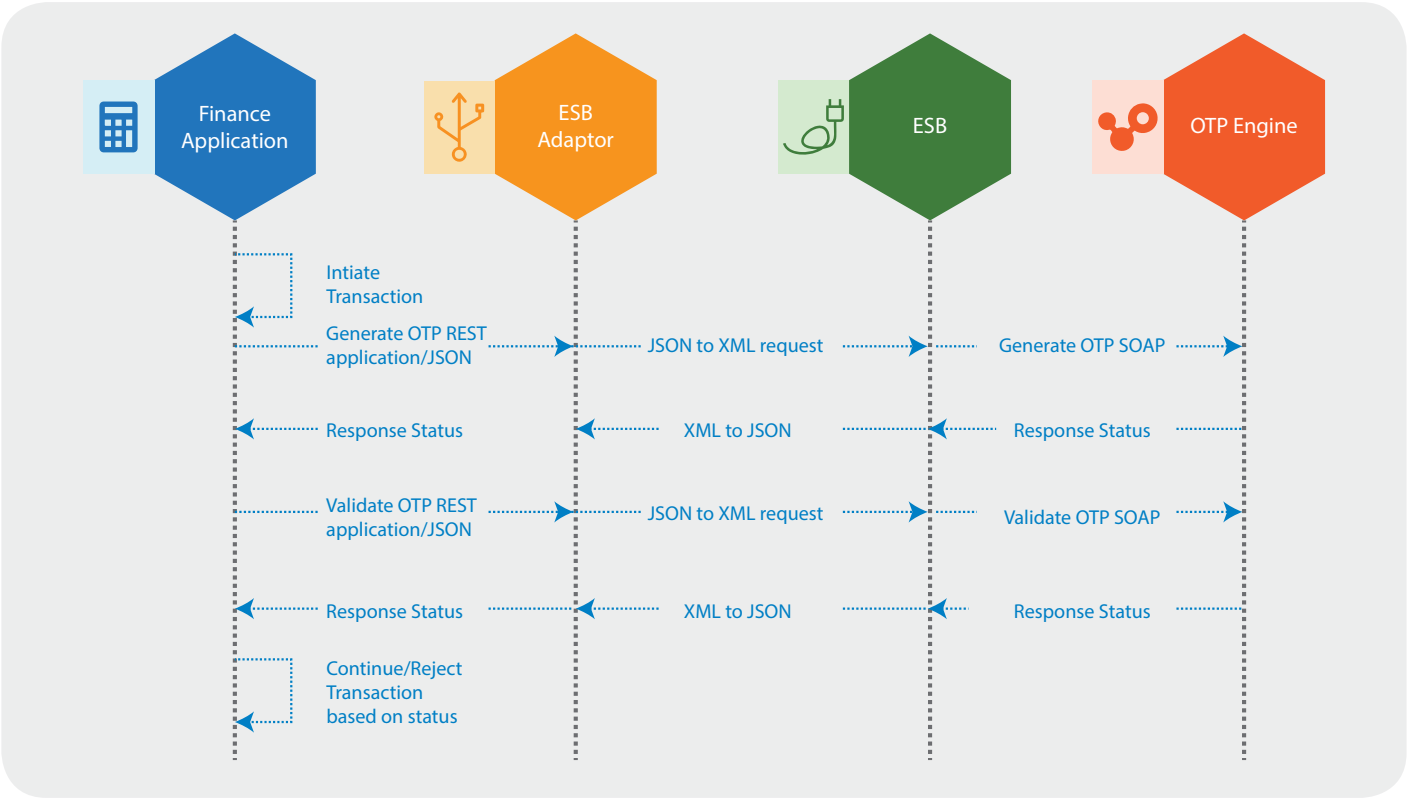
Feature	Benefit
Risk Analysis	The implementation helps in analyzing the risk associated with a user or transaction based on the risk policies configured in adaptive authentication system. Based on the risk analysis, the transaction can be allowed, reviewed, challenged or denied.
Reusability of authentication mechanism	The implementation helps in leveraging the authentication mechanisms built as a standalone entity for adaptive authentication. The AA system can trigger the authentication mechanism, which can be fulfilled by the standalone enterprise authentication system of the finance institution.
Authentication and notification systems	AA system provides the step up authentication system, and would keep hold of the authentication results for future risk analysis. If a separate authentication system is used, other than inbuilt authentication mechanism of AA system, then the implementation provides an option to notify the adaptive authentication system of the authentication results.

OTP support:

One Time Password acts as a step up authentication mechanism, and enhances the security of the transaction. OTP is required to be initiated and authenticated in case of compliance perspective for step-up authentication. It can also be prompted when the risk profile of the transaction is high. Only after validating the OTP, the user can proceed with the transaction.

For OTP to be generated and validated, it is advisable to go with a separate enterprise OTP server. As OTP can be generated not only by portal, but can also be required for various other applications including switch in an enterprise bank operations. So instead of directly accessing the OTP server, it is advisable to access the OTP server through Enterprise Service Bus

channel. If OTP server expose its services as SOAP services, and you would want your portal to consume it as REST services for uniformity and consistently, then you can build an adaptor in ESB to convert the JSON request to SOAP request, and SOAP response to JSON response. The sample flow is depicted in following diagram:



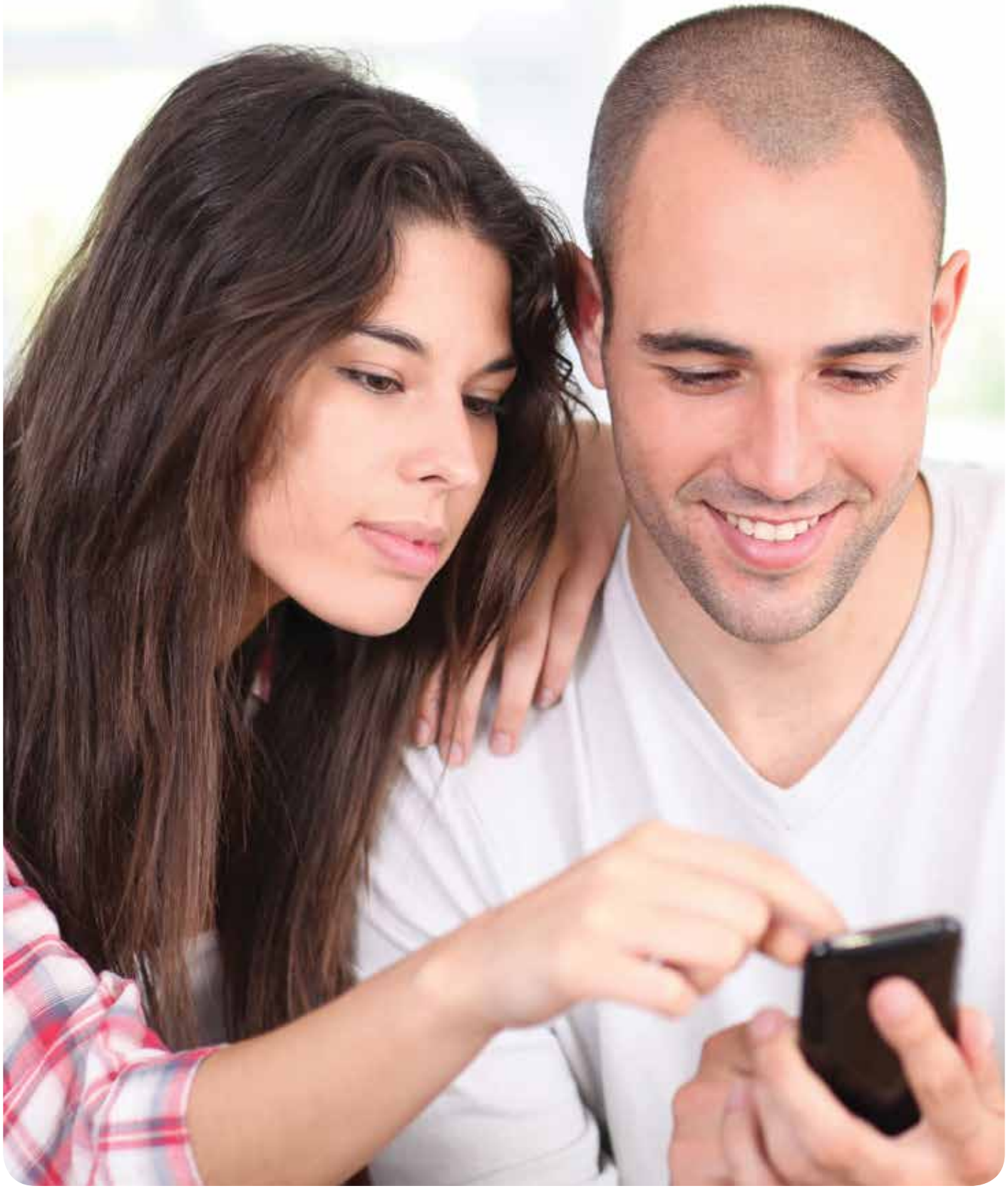
Security best practices

Main security best practices for finance applications are given below

- Identify and classify data: Classify the sensitive data as per the regulatory compliance
- Protect your data at rest:
  - If the data is stored on a drive, either the entire drive can be encrypted or file by file encryption can be done.
  - If data is stored in a database, either the entire database can be encrypted or value-by-value encryption can be done
- Protect your data in motion:
  - SSL/TLS encryption must be used for any HTTP traffic with a validate certificate from a trusted authority.
- Security events logging:It is recommended to meet the following Security Events Monitoring requirements:
- Security events must be enabled for all type of access (e.g. console, remote, terminal, etc)
- Logging configured to provide at minimum details of the user, date,time, IP Address and event.
- Log entries must be sent to remote Central log server.
- Logs must be reviewed by qualified operations staff and reported to business owner for abnormal activity.
- Session should be invalidated after log out.
- SQL Injection should not happen from the application. Application should restrict the special characters that are dangerous for SQL injection.
- Link injection should not happen from the application. Special characters should be restricted, which are prone for HTML script injection.
- Click jacking should be prevented by setting the browser response header – X-Frame-Options to DENY, that instruct the browser to not allow framing.
- The application should not be prone to Cross Site scripting. It is possible to gain access to gain access by injecting special characters in between the URL, which may allow a hacker to view or alter the user records. So, application should restrict the special characters that are dangerous for cross site scripting.
- All cookies should be set to Secure in order to prevent cookie being sent unencrypted.
- When the session is idle for a prescribed time, the session should time out and expire.
- Always change password should be allowed, if he gives the old password correctly.
- The password policy should adhere to complex password with alpha numeric, special character and a minimum length of eight. This way password cannot be predicted using brute force attack/ dictionary attack.
- Use salted hash password for authentication to avoid brute force attack.
- Application code should never be displayed on the browser through stack trace when there is an application error.
- All input data should be validated both at the client and server level.
- Always mask the sensitive information, when displayed on portal browser or logs.

Case Study

In this section we will look at a case study of a retail banking case study to discuss the security aspects



Problem Statement

XYZ bank wants to start its retail banking operations in a short timeframe, and its retail team have developed all the core features and rolled out their portal for sign off to its internal security team. But the XYZ's security team noticed many security flaws, though the portal is fully operational with all its features. Few problems noticed by security team includes

- Password is hacked and compromised, though it is encrypted.
- Able to hijack the session with predictable session identifiers.
- Able to call backend services directly hitting the service URLs without any authentication.
- Simultaneous login allowed from different browsers and devices.
- Able to go back, forward and refresh the browser, resulting in resubmission of forms.
- The portal can be accessed from same login from Namibia and India geolocation within a time frame of 15 minutes, though it is not possible to travel between two destinations in that timeframe. Remote desktops are possible from India to Namibia, but things cannot be assumed and security be compromised.
- Browser HTML code can be edited through developer tools, and can skip the div which prompts for OTP and its authentication.

Now, the retail team was quick enough to hire a security consultant, and guide the retail portal team to bridge all the flaws. Now all onus lies with this security consultant to solve all security flaws, and make XYZ Bank Go-live. Now list down what all details the newly hired security consultant should gather, and what recommendations should he give to the team.

Solution

Rather than throwing out, what he knows about all security measures and best practices the industry follows, the security consultant was interested in understanding the current system, to come out with the customized solution for the bank. The consultant heard the following things which the retail portal team had implemented.

- The retail banking portal relies on username and password validation for giving access to the retail portal. There was strong complex password policy already in place.
- The password was encrypted with SHA algorithm which is a strong encryption algorithm.
- Team has implemented security measures to restrict special characters on all form fields, to avoid SQL and XSS attack.
- The site is secure, and all sensitive information flows through HTTPS secure encryption.
- All cookies are set to secure, and cross site frames are denied.

Though the team made its best effort to adhere to all security measures, the consultant felt that there are problems on authentication mechanism implementation, which need to be supplemented with another security mechanism for session maintenance. Consultant's suggestion goes as below.

User credential validation are not enough for banking portal, and need to be complemented with a Unique Identifier (UID) authentication. He suggested to generate the UID on successful login, and store the UID in user's browser client, and server cache, holding the mapping between customer id and UID. For subsequent request from the client browser, all requests should go to the

same UID need to be sent for validation to sessionUIDFilter, and again on validation a new UID is generated and stored in user's client browser and server cache for the customer. The cycle keeps on repeating. If UID does not match, session should expire.

When user refreshes, goes back and forward, obviously UID expected by the server mismatches, and session expires. To their surprise, the same implementation also solved multiple other issues they faced. It prevented simultaneous login, as UID cannot match and new session cannot be created. It also prevented from accessing the portal service urls directly, as it goes to the sessionUIDFilter and with no valid session and right UID, the service url cannot be accessed.

Though SHA password cannot be decrypted, it can be prone to dictionary or brute force attack using lookup table. So he suggested to implement salted hash password, using BCryptPasswordEncoder. The team implemented the same using spring security.

He also suggested to implement adaptive authentication mechanism, which will track the geolocation from where the portal is accessed, and prompt for second factor authentication like OTP or security questions. The team implemented this, and helped in exceeding the expectations of security team by triggering second factor authentication for unusual behaviour of the user.

The consultant suggested to put validations and authentication checks in the server side also, so that it cannot be skipped by tweeking the HTML code. He also suggested to disable right clicks and launching developer tools from the browser.

XYZ bank again went for sign off from it security team. Now the security team is happy that they were not able to guess or lookup any of the passwords, could not hijack the session, could not have multiple simultaneous sessions, could not access service URL directly or skip authentications. It also triggered second factor authentication for suspicious behaviors. All browser navigations like back, forward and refresh expired the session. Sign off was given and XYZ bank went live without any security problems. The consultant did his piece of work fabulous.

About the Authors

Shailesh Kumar Shivakumar

Shailesh Kumar Shivakumar has over 14 years of industry experience. He is currently working as Senior Technology architect at Digital Practice in Infosys technologies. His areas of expertize includes Enterprise Java technologies, Portal technologies, Web Technologies, Performance Engineering. He has published two books related to enterprise web architecture and Enterprise Portals and UXP and has four patent applications. He has published several papers and presented talks in IEEE conferences in the areas of web technologies and performance engineering. He has successfully lead several large-scale enterprise engagements for Fortune 500 clients.

Babu Krishnamurthy

Babu Krishnamurthy has over 11 years of industry experience. He is currently working as Technology architect at Digital Practice in Infosys technologies. His domain expertise include banking, ecommerce and networking domain. His area of expertise in technology includes Enterprise Java technologies, Portal technologies, eCommerce technologies and Web Technologies.