Two factor authentication (2FA) with Servoy

2FA is an extra step in the authentication process. After submitting username / password, a token is generated and sent to the user by email or SMS. The user needs to enter the token in the login screen.

Normally you would log the token in a separate table together with device and OS information. You can use this log for preventing 2FA from the same device that was earlier verified for the user (seems that facebook works that way).

The 2FA functionality can easily be implemented in existing login functions.

Screens / emails

Login screen

Email with pin code

Enter pincode

Setting up the table

Use servoy to create a table (dbi) in your DB (NOT in svy-security) with the following specs:

Name	Type	Length	Null	PK
Id	INTEGER		Not	pk
userId	INTEGER		Not	
Token	TEXT	36		
os_name	TEXT	20		
ip_address	TEXT	30		
user_agent	TEXT	500		
pin_verified_date	DATETIME			
last_login_date	DATETIME			

Scripts (do not copy/paste because @porperties get mixed up)

Login.js

```
/**
* @properties={typeid:35,uuid:"120A9B78-AF38-46A0-94FC-9A7D1FB95370",variableType:-4}
*/
var user_agent = plugins.ngclientutils.getUserAgent();
/**
* @properties={typeid:35,uuid:"D36EE3A9-A0A5-4D52-8D40-603E06F71E35",variableType:-4}
*/
var ip_address = application.getIPAddress();
/**
* @properties={typeid:35,uuid:"BA79A173-B98A-4F1E-90B9-741D6274C379",variableType:-4}
*/
var os_name = application.getOSName();
/**
* @type {String}
*
* @properties={typeid:35,uuid:"114D2A5D-EE59-4A2E-9CA9-63BCF6969E79"}
*/
var pinCode;
/**
* @param firstShow
* @param event
*
* @properties={typeid:24,uuid:"7E93882E-992F-4BA3-8B5A-B2C522BB4047"}
* @override
*/
function onShow(firstShow,event) {
_super.onShow(firstShow,event);
tenantName = "Test Tenant";
```

```
onLoginSuccess();


}
/**
* @param error
*
* @properties={typeid:24,uuid:"CC3B6B90-A9CF-4EF8-B07C-08C93EB32BF3"}
* @override
*/
function onLoginError(error) {
plugins.webnotificationsToastr.error(error);
elements.labelError.visible = true;
}
/**
* @properties={typeid:24,uuid:"76C772F1-3B71-429C-A324-4F66A3FD07FF"}
* @override
*/
function onLoginSuccess() {
elements.labelError.visible = false;
}
/**
* @properties={typeid:24,uuid:"11CCA252-722A-4A86-9DC9-249C7F55A8FC"}
* @override
*/
function login() {
if(!tenantName){
onLoginError(ERROR_CODES.TENANT_NOT_SPECIFIED);
return false;
}
if(!userName){
onLoginError(ERROR_CODES.USER_NOT_SPECIFIED);
return false;
}
if(!password){
onLoginError(ERROR_CODES.PASSWORD_NOT_SPECIFIED);
return false;
}

var tenant = scopes.svySecurity.getTenant(tenantName);
if(!tenant){
```

```
onLoginError(ERROR_CODES.TENANT_NOT_FOUND);
return false;
}
var user = tenant.getUser(userName);
if(!user){
onLoginError(ERROR_CODES.USER_NOT_FOUND);
return false;
}
if(!user.checkPassword(password)){
onLoginError(ERROR_CODES.PASSWORD_MISMATCH);
return false;
}
if(user.isLocked()){
onLoginError(ERROR_CODES.LOCKED_USER);
return false;
}
if(tenant.isLocked()){
onLoginError(ERROR_CODES.LOCKED_TENANT);
return false;
}
// check for tenant into IPS Database
var ipsTenantID = scopes.ipsSecurity.getIPSTenantID(tenantName);
if (!ipsTenantID) {
onLoginError(ERROR_CODES.TENANT_NOT_FOUND);
return false;
}

// check for ipsuser into IPS Database
var ipsUserID = scopes.ipsSecurity.getIPSUserID(userName, ipsTenantID);
if (!ipsUserID) {
onLoginError(ERROR_CODES.USER_NOT_FOUND);
return false;
}

// check tenants table if 2FA is on
var do_two_fa = scopes.ipsSecurity.getIPSdo2FA(tenantName); // check tenant for 2FA
if (do_two_fa) {
if (checkPin(user, ipsUserID)) {
saveLastLogin(ipsUserID);
} else {
```

```
return false;
}
}

if(!scopes.svySecurity.login(user, ipsUserID + "")){
onLoginError(ERROR_CODES.INSUFFICIENT_PERMISSIONS);
return false;
}
// set session variables for logged user
scopes.ipsSecurity.initIPSSession(ipsUserID, ipsTenantID)

// notify success
onLoginSuccess();

return true;
}
/**
* @param user {User}
* @param userId
*
* @properties={typeid:24,uuid:"0624F312-101B-4FA2-88F3-2B41582CFE23"}
*/
function checkPin(user, userId) {
  var qry = datasources.db.ips.ipsuser_pin.createSelect();
  qry.result.add(qry.columns.userid);
  qry.result.add(qry.columns.token);
  qry.result.add(qry.columns.ip_address);
  qry.result.add(qry.columns.os_name);
  qry.result.add(qry.columns.user_agent);
  qry.result.add(qry.columns.pin_verified_date);
  qry.where.add(qry.columns.userid.eq(userId));
  qry.where.add(qry.columns.user_agent.eq(user_agent));
  qry.where.add(qry.columns.os_name.eq(os_name));

  var fs = databaseManager.getFoundSet(qry);
  fs.loadAllRecords();

  if (fs.getSize() > 0) {
    if (fs.pin_verified_date && fs.pin_verified_date <= new Date()) { // token has been verified
      return true;
```

```
    } else { // token not (yet) validated
    if (pinCode) {
       if (actionPin(userId, pinCode, 'validate')) { // token is valid...set / save pin_verified_date
         actionPin(userId, pinCode, 'veryfied');
         return true;
        } else {
         // remove from ipsuser_pin
         actionPin(userId, pinCode, 'delete');

         // reset form fields
         userName = null;
         password = null;
         pinCode = null;

         updateUI(); // reset UI to user / password
         return false;
        }
    } else {
     updateUI('token');
     return false;
    }
    }
} else {
    var token = user.generateAccessToken();
    var pin = token.substring(0, 6);

    fs.newRecord();

    // save to DB
    fs.userid = userId;
    fs.ip_address = ip_address;
    fs.user_agent = user_agent;
    fs.os_name = os_name;
    fs.token = token;

    databaseManager.saveData(fs);
    var mobile = scopes.ipsSecurity.getIPSUserSMS(userId); // get mobile number for SMS messages
    if (!mobile || mobile.length == 0) { // send email with token
     var emailSubject = i18n.getI18NMessage('ips.login.email_key_subject');
     var emailMessage = i18n.getI18NMessage('ips.login.email_key_body', [user.getDisplayName(), pin]);
```

```
        scopes.ipsSMTPEmail.onSMTPEmail(userName, emailSubject, emailMessage);

        plugins.webnotificationsToastr.success(i18n.getI18NMessage('ips.login.email_validate_pin'));
       } else { // send via SMS

        var ipsCountry = i18n.getCurrentCountry()
        if (ipsCountry != 'AU') { // use UK sms provider
       sendSMSuk(mobile, pin);
        } else { // user AUS sms provider
        sendSMSaus(mobile, pin);
        }

        plugins.webnotificationsToastr.success(i18n.getI18NMessage('ips.login.sms_validate_pin'));
       }

       updateUI('token');
       return false;
   }
}
/**
* @param mobile {String}
* @param pin {string}
*
* @properties={typeid:24,uuid:"6643FA51-8D4D-417B-B353-DCC9FEA2A91A"}
*/
function sendSMSuk(mobile, pin) {
var smsToken = '<<<token from SMS provider>>>';
var creditsURL = 'https://my.fastsms.co.uk/api?Token=' + smsToken + '&Action=CheckCredits';

var client = plugins.http.createNewHttpClient();
var request = client.createGetRequest(creditsURL);
var response = request.executeRequest();

if (parseFloat(response.getResponseBody()) > 0) { // credits left -> send SMS
var msg = 'Your PIN Code to the Turnkey Insolvency Creditor Portal is ' + pin + ' Please do not reply to this message.';

var msgURL = 'https://my.fastsms.co.uk/api?Token=' + smsToken +
'&Action=Send&DestinationAddress=' + mobile +
'&SourceAddress=I<<<Your name>>>PS-DOCS&Body=' + encodeURI(msg);
```

```
request = client.createGetRequest(msgURL);
response = request.executeRequest();
}
}
/**
* @param mobile {String}
* @param pin {string}
*
* @properties={typeid:24,uuid:"1C18586D-9A02-4214-8EBD-CCD1CF6AB76A"}
*/
function sendSMSaus(mobile, pin) {
var smsUser = '<<<Your name>>>>';
var smsToken = '<<<token from SMS provider>>>';
var creditsURL = 'https://api-mapper.clicksend.com/http/v2/balance.php?username=' + smsUser +
'&key=' + smsToken;

var client = plugins.http.createNewHttpClient();
var request = client.createGetRequest(creditsURL);
var response = request.executeRequest();
var xml = response.getResponseBody();
var re = new RegExp('<balance>([^<]*?)<', 'gi');
var balance = re.exec(xml);

if (parseFloat(balance[1]) > 0) { // credits left -> send SMS
var msg = 'Your PIN Code to the Turnkey Insolvency Creditor Portal is ' + pin + ' Please do not reply to this message.';

var msgURL = 'https://api-mapper.clicksend.com/http/v2/send.php?method=http&username=' + smsUser +
'&key=' + smsToken +
'&to=' + mobile +
'&message=' + encodeURI(msg);

request = client.createGetRequest(msgURL);
response = request.executeRequest();

application.output('cs');
}
}
/**
* @param userId {Integer}
* @param pin {String}
```

```
* @param action {String}
* @properties={typeid:24,uuid:"E1DC04CE-5A90-4B37-882D-261D5E2A3CE8"}
*/
function actionPin(userId, pin, action) {
// get token based on userId and pin
   var qry = datasources.db.ips.ipsuser_pin.createSelect();
   qry.result.add(qry.columns.token);
   qry.result.add(qry.columns.pin_verified_date);
   qry.where.add(qry.columns.ipsuserid.eq(userId));
   qry.where.add(qry.columns.token.substring(1, 6).eq(pin))

/** @type {JSFoundset<db:/ips/ipsuser_pin>} */
var fs = databaseManager.getFoundSet(qry);
   fs.loadAllRecords();

switch(action) {
case 'validate':
if (fs.getSize() > 0) {
      if (scopes.svySecurity.consumeAccessToken(fs.token)) {
       return true;
      } else {
       return false;
      }
  }
   return false;
   break;
case 'veryfied':
fs.pin_verified_date = new Date();
databaseManager.saveData(fs);
break;
case 'delete':
fs.deleteRecord();
break;
}
}
/**
* @param [updtok]
*
* @properties={typeid:24,uuid:"0E51EA43-7932-4D67-A994-6020B41287B8"}
*/
```

```
function updateUI(updtok) {
    if (updtok) { // set input to validate token
        elements.fontawesomeUser.visible = false;
        elements.textboxUser.visible = false;
        elements.fontawesomePassword.visible = false;
        elements.textboxPassword.visible = false;

        elements.fontawesomeToken.visible = true;
        elements.textboxToken.visible = true;
    } else { // set input to enter username / password
        elements.fontawesomeUser.visible = true;
        elements.textboxUser.visible = true;
        elements.fontawesomePassword.visible = true;
        elements.textboxPassword.visible = true;

        elements.fontawesomeToken.visible = false;
        elements.textboxToken.visible = false;
    }
}
/**
* @param userId {Integer}
*
* @properties={typeid:24,uuid:"0A3607D3-9732-4B9F-BEAA-EDF4A604B104"}
*/
function saveLastLogin(userId) {
    var qry = datasources.db.ips.ipsuser_pin.createSelect();
    qry.result.add(qry.columns.id);
    qry.result.add(qry.columns.last_login_date);
    qry.where.add(qry.columns.ipsuserid.eq(userId));
    qry.where.add(qry.columns.user_agent.eq(user_agent));
    qry.where.add(qry.columns.os_name.eq(os_name));

/** @type {JSFoundset<db:/ips/ipsuser_pin>} */
var fs = databaseManager.getFoundSet(qry);
    fs.loadAllRecords();

    fs.last_login_date = new Date();
    databaseManager.saveData(fs);
}
/**
```

```
* @param {JSEvent} event
* @param {string} dataTarget
*
* @protected
*
* @properties={typeid:24,uuid:"21F86273-458C-42E9-89BB-A2A3AEE2749A"}
*/
function onActionRequestNewPassword(event, dataTarget) {
var user = scopes.svySecurity.getUser(userName);

// generate temp token
  var token = user.generateAccessToken();

  // send email with token
  var emailSubject = i18n.getI18NMessage('ips.login.email_reset_subject');
  var emailMessage = i18n.getI18NMessage('ips.login.email_reset_body', [user.getDisplayName(), token]);

  scopes.ipsSMTPEmail.onSMTPEmail(userName, emailSubject, emailMessage);

  plugins.webnotificationsToastr.success(i18n.getI18NMessage('ips.login.email_reset_toastr'));

  application.showForm('ipsLoginReset')
}
```

ipsSecurity.js (scopes.ipsSecurity)

```
/**
* @protected
* @type {String}
*
* @properties={typeid:35,uuid:"B212C55F-1953-466D-938F-A2FF6785A7AB"}
*/
var loggedTenantID;
/**
* @protected
* @type {String}
*
* @properties={typeid:35,uuid:"8DE5C0AA-87CD-49D9-AB6E-22044B49C6D7"}
*/
var loggedUserID;
/**
* Retrieve the tenant name based on the unique username.
```

```
* Throws an exception if the username is not unique.
* @param {String} userName
*
* @return {String}
*
* @throws {scopes.svyExceptions.IllegalStateException}
*
* @properties={typeid:24,uuid:"BAE2D138-0636-4969-9ED1-858A8F3DE65D"}
*/
function getTenantName(userName) {
var query = datasources.db.svy_security.users.createSelect();
query.result.clear();
query.result.add(query.columns.tenant_name)
query.where.add(query.columns.user_name.eq(userName));
var ds = databaseManager.getDataSetByQuery(query, -1);
if (ds.getMaxRowIndex() == 1) {
return ds.getValue(1, 1);
} else if (ds.getMaxRowIndex() == 0) {
return null;
} else if (ds.getMaxRowIndex() > 1) {
throw new scopes.svyExceptions.IllegalStateException("this should not happen, there should be only 1 username globally");
}
return null;
}
/**
* Return the Master Tenant object
* @public
*
* @return {scopes.svySecurity.Tenant}
*
* @properties={typeid:24,uuid:"21952832-C0C6-4961-87E6-2E3B156A65B8"}
*/
function getMasterTenant() {
return scopes.svySecurity.getTenant('tenant');
}
/**
* @public
* @param {String} userName
* @param {Number} [tenantID]
*
```

```
* @return {Number}
*
* @properties={typeid:24,uuid:"874647FB-B1D1-492F-A25E-B322A7845980"}
*/
function getIPSUserID(userName, tenantID) {
var query = datasources.db.ips.ipsuser.createSelect();
query.result.addPk();
query.where.add(query.columns.logonname.eq(userName));
query.where.add(query.columns.tenantid.eq(tenantID));

var ds = databaseManager.getDataSetByQuery(query, 1);
if (ds.getMaxRowIndex()) {
return ds.getValue(1,1);
}
return null;
}
/**
* @public
* @param {String} tenantName
* @return {Number}
*
* @properties={typeid:24,uuid:"20648A3B-25D4-4D46-8190-D165B6685EB1"}
*/
function getIPSTenantID(tenantName) {
var query = datasources.db.ips.tenant.createSelect();
query.result.addPk();
query.where.add(query.columns.name.eq(tenantName));

var ds = databaseManager.getDataSetByQuery(query, 1);
if (ds.getMaxRowIndex()) {
return ds.getValue(1,1);
}
return null;
}
/**
* @public
* @param {String} tenantName
* @return {Number}
*
*
```

```
* @properties={typeid:24,uuid:"83CA1364-8537-44FA-A7F2-666DDF699966"}
*/
function getIPSdo2FA(tenantName) {
var query = datasources.db.ips.tenant.createSelect();
query.result.add(query.columns.do_two_fa)
query.where.add(query.columns.name.eq(tenantName));

var ds = databaseManager.getDataSetByQuery(query, 1);
if (ds.getMaxRowIndex()) {
return ds.getValue(1,1);
}
return null;
}
/**
* @public
* @param {Number} userId
*
* @return {String}
*
* @properties={typeid:24,uuid:"2177C423-5848-4C0C-A794-B7A438060582"}
*/
function getIPSUserEmail(userId) {
   var query = datasources.db.ips.ipsuser.createSelect();
   query.result.add(query.columns.email);
   query.where.add(query.columns.id.eq(userId));

   var ds = databaseManager.getDataSetByQuery(query, 1);
   if (ds.getMaxRowIndex()) {
      return ds.getValue(1,1);
   }
   return null;
}
/**
* @public
* @param {Number} userId
*
* @return {String}
*
* @properties={typeid:24,uuid:"F4D18D0E-7286-4FDC-B9D6-F1ACAF6D585A"}
*/
```

```
function getIPSUserSMS(userId) {
   var query = datasources.db.ips.ipsuser.createSelect();
   query.result.add(query.columns.smsnumber);
   query.where.add(query.columns.id.eq(userId));

   var ds = databaseManager.getDataSetByQuery(query, 1);
   if (ds.getMaxRowIndex()) {
      return ds.getValue(1,1);
   }
   return null;
}
/**
* @public
* @param {Number} userID
* @param {Number} tenantID
*
* @properties={typeid:24,uuid:"4B536651-BC97-4E21-9177-312EE253BAAA"}
*/
function initIPSSession(userID, tenantID) {
loggedTenantID = tenantID;
loggedUserID = userID;
}
/**
* @public
* @properties={typeid:24,uuid:"7E7702F4-B7FE-4F7F-BDAE-6D3623E725DE"}
*/
function getLoggedUserID() {
return loggedUserID;
}
/**
* @public
* @properties={typeid:24,uuid:"7BA15650-F295-4B1D-BCD3-953ED6088705"}
*/
function getLoggedTenantID() {
return loggedTenantID;
}
```

Gerelateerde artikelen

- Two factor authentication (2FA) with Servoy