SparkView

Integration Quick Start

1.0
August 27, 2016

1. Connect to a RDP server using URL parameters

Save a JavaScript file as index.page.js with the following content on your web server:

```
window.onload = function() {
        var gateway = '192.168.12.111',//change this to your Spark gateway address
            server = '192.168.12.117',//change this to your RDP server address
            url = 'ws://' + gateway + '/RDP?server=' + server +
                '&user=userName&pwd=password';
      var r = new svGlobal.Rdp(url);
       r.addSurface(new svGlobal.LocalInterface());
      r.run();
};
```

You can use JSP, ASP.net, PHP or other server side programming to get the server, credentials from your database.

Save a web page as index.html with the following content on your web server:

```
<!doctype html>
<html>
<head>
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Spark View (RDP)</title>
<meta name="viewport" content="width = device-width, initial-scale = 1.0, user-scalable =
yes, minimum-scale = 0.1, maximum-scale = 8" />
```

```
<meta name="apple-mobile-web-app-capable" content="yes" />


<link rel="stylesheet" href="../hi5.css" />
<link rel="stylesheet" href="../rdp.css" />


<script type="text/javascript" src="../appcfg.js"></script>
<script type="text/javascript" src="../resource.js"></script>
<script type="text/javascript" src="../hi5_min.js"></script>
<script type="text/javascript" src="../surface_min.js"></script>
<script type="text/javascript" src="../rdp_min.js"></script>
<script type="text/javascript" src="index.page.js"></script>
</head>


<body>
    <div>
        <canvas id="remotectrl"></canvas>
    </div>
</body>
</html>
```

Pros: easy and simple

Cons: user can see the URL by checking the Developer tool of browser.

2. Connect to a RDP server using Cookies

Modify your index.page.js as the followings:

```
window.onload = function() {
        document.cookie = "gateway=192.168.12.111' ";
        document.cookie = "server=192.168.12.117";
        document.cookie = "user=userName";
        document.cookie = "pwd=password";
        var r = new svGlobal.Rdp2();
         r.addSurface(new svGlobal.LocalInterface());
        r.run();
};
```

You should use server side technologies (JSP, ASP.net, PHP etc) to save the cookie.

Pros: Parameters are saved in the cookies and user will not see it in the URL.

Cons: User's browser may have cookies disabled. User can still see the parameters by checking the value of the cookies.

3. Setting up parameter with HTTP header or HTTP Basic Authentication for VPN integration

Spark View also supports the following parameters from HTTP Headers:

gw_server: The RDP server address.

gw_port: RDP server port

gw_user: username of RDP server

gw_pwd: password of RDP server

gw_symlink: the symlink id

You can also set authorization = Basic in gateway.conf to enable HTTP Basic Authentication on Spark Gateway.

Pros: easy and simple for SSO

Cons: Need VPN or proxy to set up the HTTP headers.

4. Connect to a RDP server with symlink

You need to create the server and symlink first on the gateway.

You can use config.html or HTTP API to create a server on the gateway:

HTTP API:

http://gatewayAddress/SERVER?id=serverId&displayName=Name&server=192.168.1.117&gatewayPwd=21232f297a57a5a743894a0e4a801fc3&...

You only need to do this one and you can put other parameters into this URL, such user, pwd, performanceFlags etc.

Every time after user logged into your portal, you create a temporary symlink for this user using HTTP API:

http://gatewayAddress/SYMLINK?symlink=symlinkId&server=serverId&validTime=5m&gatewayPwd=passwordInGateway.conf&...

gatewayPwd is hexadecimal MD5 hash of the password which is configured in gateway.conf.

This symlink will be invalid after 5 minutes.

You can also put other RDP parameter when creating the symlink, for example:

```
var url =
'http://gatewayAddress/SERVER?id=serverId&displayName=Name&server=192.168.1.117&gatewayPwd=21232f297a57a5a743894a0e4a801fc3&parameters=' +
encodeURIComponent('user=' + userName + '&pwd=' + password);
```

Then you let the user connect to this symlink instead:

```
window.onload = function() {
        var gateway = '192.168.12.111',//change this to your Spark gateway address
        symlinkId = 'theSymlinkId',
            displayName = 'serverName',

            url = 'ws://' + gateway + '/RDP?symlink=' + symlinkId + + '&displayName=' +
                displayName;

        var r = new svGlobal.Rdp(url);

         r.addSurface(new svGlobal.LocalInterface());

        r.run();

};
```

Expired symlink will not affect connected sessions.

Pros: server address and credentials are saved on the gateway side. User can only see a symlink id which will be invalid in a short time. This method works with VPN or without VPN.

Cons: You may need to recreate the symlink for user if the symlink expired.

5. Connect to a RDP server with a Spark Gateway plug-in

With a Spark Gateway plug-in, you can:

1. Encrypt the parameter in your portal and send it to the gateway; the gateway plug-in decrypt the data.
2. Verify the connection, then refuse or let the connection go throght.
3. Control almost everything on the gateway: list, terminate sessions, control resource redirections on the fly etc.

Pros: do almost anything you want.

Cons: need to write some simple code in Java.

Please check the Integration Guide and the plug-in example for more details:

http://remotespark.com/view/IntegrationGuide.pdf

http://remotespark.com/Plugin.zip