Integration of Office 365 with existing faculty SSO

Best Practice Document

Produced by the MARnet-led working group on campus wireless infrastrucure and security

Authors: Vasko Sazdovski (MARnet), Boro Jakimovski (MARnet).

March 2015

Document No:	GN3plus-NA3-T2-CBPD MA3
Version / date:	March 2015
Original language	Macedonian
Original title:	“Интеграција на Office 365 со постоечки систем за единствена најава”
Original version / date:	Version 1.0; 1 October 2014
Contact:	vasko.sazdovski@finki.ukim.mk

MARnet bears responsibility for the content of the document. The work has been carried out by a MARnet led working group Campus wireless infrastructure and security.

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 605243, relating to the project 'Multi-Gigabit European Research and Education Network and Associated Services (GN3plus)'.

Table of Contents

Table of Figures

Executive Summary

How to integrate Office 365 with your local SSO (Single Sign-On)? This question and all steps that are necessary to integrate this solution are described in this document. The idea for this integration derives from the problem with multiple accounts. The following Sections show step-by-step the whole integration process – what you need to implement and to install in order to provide this service.

Section 1 presents the best practice architecture to use in faculty and/or university environments.

Section 2 describes the implementation process, the details and attributes that are needed, so one can implement SSO in your organisation in preparation for Office 365 integration.

Section 3 describes how one can federate an existing Office 365 domain in order to be able to accept authentication messages from a local SSO.

Section 4 describes the final step of user synchronisation tools that must be completed in order to link local users with the accounts in Microsoft Office 365.

1 Architecture

The architecture of this system is at a medium level of complexity and combines different technologies. Microsoft AD, PowerShell, Shibboleth IdP, JASIG CAS and Open LDAP are all combined.

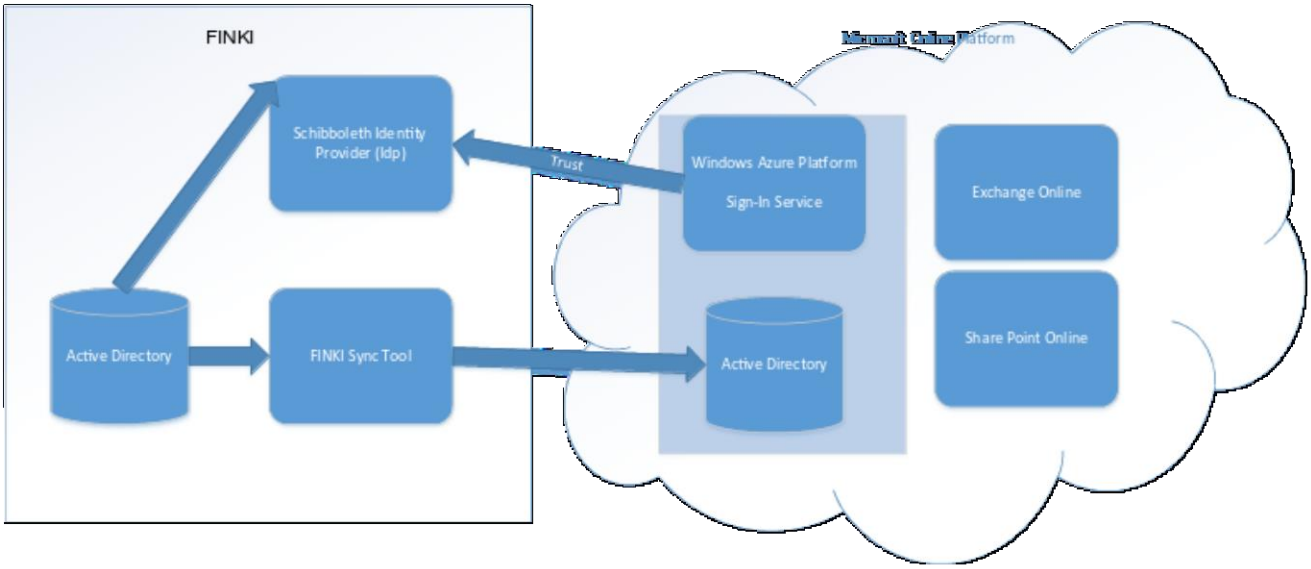

Figure 1.1: Integration of Office 365 with existing faculty SSO based on Active Directory and Shibboleth

1.1 Local user account management

These days the use of email service has become part of our daily routine. In order to make the communication with students much easier and official, your faculty can create a subdomain for the students. The best solution was the Live@Edu service from Microsoft: it offered an API that was easy to use and implement with our SSO. Using this, all of the students were able to login to their email accounts using the existing SSO.

After Microsoft migrated from Live@Edu to Office 365, they abandoned the API, and another solution was required to provide email service for students. Microsoft suggested the use of ADFS, so all local users could be synchronised with Azure AD. This solution is complex and it needed the deployment of many tools and services that in order to establish inter-federation, so that Office 365 services could be used with your local user accounts. An alternative solution is to deploy SAML2 (i.e. Shibboleth) IdP and to establish federation with Microsoft, so that you could provide SSO access to Office 365 services with your existing SSO system based on SAML2. This is less challenging solution to implement, and is recommended for all institutions that do not keep local user accounts in Active Directory.

We prefer to keep local student's accounts in an on-site Microsoft AD (Active Directory), since this offers good integration and easier management with locally provided MS services, such as login to

Laboratory computers based on MS Windows. In the scenarios that follow, this information is taken as a starting point from where the whole architecture built on. Nevertheless one can still build an Office 365 integrated SSO using SAML2, even if the user accounts are not kept in Microsoft AD, but in some other technologies like LDAP, or HR databases.

The first step in the integration process is deploying an SAML2 IdP (Shibboleth IdP) as an Identity Provider that is integrated with the local users' database (Microsoft AD) for authentication. After deploying the IdP, the users domain has to be federated in order to establish the SSO.

1.2 Local IdP

The local Identity Provider (IdP) is installed on a Debian server where the Shibboleth IdP is used as an Identity Provider. The IdP is connected to a JASIG CAS instance as an external authentication service. The deployment scenario is is a result of JASIG CAS having been used for over eight years as an organisational SSO system for the existing local services. Since Shibboleth supports external authentication, integrating these two is easy.

Before deploying IdP, all the local web applications that were used by the students and the faculty staff were using CAS. This solution provided SSO, but the problem with this is that CAS provides only authentication, and there had to be some other mechanism to provide the authorisation attributes to the applications. After successful authentication, CAS returns the authenticated username, but none of the other attributes. In this way, the faculty had to deploy an additional LDAP for the applications to retrieve the necessary attributes. Since some of the existing services were using CAS, and because Shibboleth support CAS, we did not need to change the services that were available only for local users, and all services (local and national) remained under the legacy authentication system.

If you have an existing SSO implementation, it is possible to connect with your local instance of Shibboleth. All that is needed to be done is to modify the login handler.

Figure 1.2 shows the Identity Provider architecture.

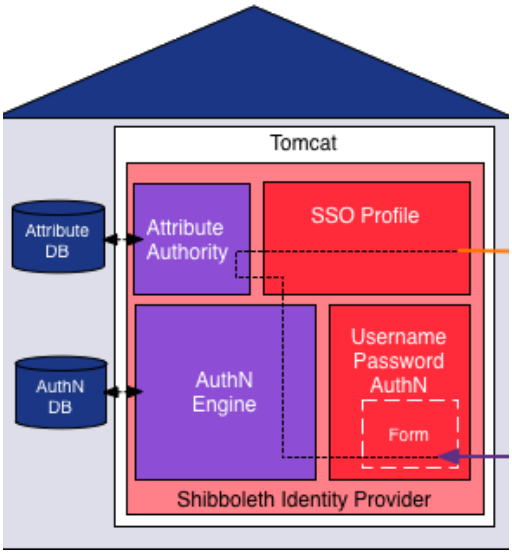

Figure 1.2: Identity Provider architecture

2 Implementation

In this Section we give the implementation details regarding federation between Shibboleth IdP and Office 365.

2.1 IDP Configuration

We begin with the configuration of local IdP that must be carried out in order to prepare the trust and attribute publishing between Local IdP and Office 365. This is necessary for successfully federatng with Microsoft, and to achieve SSO with Microsoft Office 365.

Obtaining metadata from Microsoft

Metadata is vital for federation. With this metadata, mutual trust is established between the service provider (in this case Microsoft) so it can forward the users of the federated domain for authentication. Microsoft's metadata can be obtained from the following URL:

https://nexus.microsoftonline-p.com/federationmetadata/saml20/federationmetadata.xml

Metadata is saved in the following folder in the Shibboleth IdP: /opt/shibboleth-idp/metadata, under this name: WindowsAzureAD-metadata.xml.

The content of this XML file is:

```
<?xml version="1.0" encoding="utf-8"?>
<EntityDescriptor     xmlns="urn:oasis:names:tc:SAML:2.0:metadata"
entityID="urn:federation:MicrosoftOnline">
  <SPSSODescriptor                    WantAssertionsSigned="true"
protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"
>
    <AssertionConsumerService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Location="https://login.microsoftonline.com/login.srf"   index="0"
isDefault="true"/>
```

```
    <AssertionConsumerService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST-
SimpleSign" Location="https://login.microsoftonline.com/login.srf"
index="1"/>
    <AssertionConsumerService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:PAOS"
Location="https://login.microsoftonline.com/login.srf"   index="2"
/>
    <NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-
format:emailAddress
    </NameIDFormat>
    <NameIDFormat>urn:mace:shibboleth:1.0:nameIdentifier
    </NameIDFormat>
    <NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-
format:unspecified
    </NameIDFormat>
    <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:transient
    </NameIDFormat>
    <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent
    </NameIDFormat>
  </SPSSODescriptor>
</EntityDescriptor>
```

2.2 Attributes

2.2.1 Attribute resolver

The next step is to make the necessary changes to the Shibboleth IdP **attribute-resolver.xml** that can be found in **/opt/shibboleth-idp/conf/**.

This configuration file specifies which attributes shall be resolved from our local AD (local attribute source), and is sent to Microsoft Office 365 upon successful authentication. Microsoft requires the following attributes to be available to authorise the users to use Office 365:

- Uid.
- Mail.
- CN.
- Sn.
- Domain.

- ImmutableID.

The following code is the modified content of the attribute resolver:

```
<resolver:AttributeDefinition    id="uid"    xsi:type="ad:Simple"
sourceAttributeID="uid">
   <resolver:Dependency ref="myLDAP" />
   <resolver:AttributeEncoder          xsi:type="enc:SAML1String"
       name="urn:mace:dir:attribute-def:uid" />
   <resolver:AttributeEncoder          xsi:type="enc:SAML2String"
       name="urn:oid:0.9.2342.19200300.100.1.1"
       friendlyName="uid" />
</resolver:AttributeDefinition>

<resolver:AttributeDefinition   id="email"   xsi:type="ad:Simple"
      sourceAttributeID="mail">
   <resolver:Dependency ref="myLDAP" />
   <resolver:AttributeEncoder          xsi:type="enc:SAML1String"
       name="urn:mace:dir:attribute-def:mail" />
   <resolver:AttributeEncoder          xsi:type="enc:SAML2String"
       name="urn:oid:0.9.2342.19200300.100.1.3"
       friendlyName="mail" />
</resolver:AttributeDefinition>

<resolver:AttributeDefinition id="commonName" xsi:type="ad:Simple"
      sourceAttributeID="cn">
   <resolver:Dependency ref="myLDAP" />
   <resolver:AttributeEncoder          xsi:type="enc:SAML1String"
       name="urn:mace:dir:attribute-def:cn" />
   <resolver:AttributeEncoder          xsi:type="enc:SAML2String"
       name="urn:oid:2.5.4.3" friendlyName="cn" />
</resolver:AttributeDefinition>

<resolver:AttributeDefinition  id="surname"  xsi:type="ad:Simple"
      sourceAttributeID="sn">
   <resolver:Dependency ref="myLDAP" />
   <resolver:AttributeEncoder          xsi:type="enc:SAML1String"
       name="urn:mace:dir:attribute-def:sn" />
   <resolver:AttributeEncoder          xsi:type="enc:SAML2String"
       name="urn:oid:2.5.4.4" friendlyName="sn" />
</resolver:AttributeDefinition>

<resolver:AttributeDefinition id="eduPersonOffice 365Domain"
           xsi:type="ad:Simple" sourceAttributeID="domain">
        <resolver:Dependency ref="staticEntitlements" />
```

```
        <resolver:AttributeEncoder xsi:type="enc:SAML1String"
            name="urn:mace:dir:attribute-def:eduPersonOffice
365Domain" />

    <resolver:AttributeEncoder xsi:type="enc:SAML2String"
            name="urn:oid:2.16.807.10.1.10.21"
            friendlyName="eduPersonOffice 365Domain" />
    </resolver:AttributeDefinition>

    <resolver:AttributeDefinition id="eduPersonPersistentID"
            xsi:type="ad:Simple"
            sourceAttributeID="OBJECTGUID">
        <resolver:Dependency ref="myLDAP" />
        <resolver:AttributeEncoder xsi:type="enc:SAML1String"
            name="urn:mace:dir:attribute-
def:eduPersonPersistentID" />
        <resolver:AttributeEncoder
xsi:type="enc:SAML2StringNameID"
            nameFormat="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent"
            name="urn:2.16.807.10.1.10.20"/>
    </resolver:AttributeDefinition>

    <resolver:AttributeDefinition id="eduPersonUniqueID"
            xsi:type="ad:TransientId"
            sourceAttributeID="uid">
        <resolver:Dependency ref="myLDAP" />
        <resolver:AttributeEncoder
xsi:type="enc:SAML1StringNameIdentifier"
              nameFormat="urn:mace:shibboleth:1.0:nameIdentifier"
              name="urn:mace:dir:attribute-def:eduPersonUniqueID"
           />
        <resolver:AttributeEncoder
xsi:type="enc:SAML2StringNameID"
              nameFormat="urn:oasis:names:tc:SAML:2.0:nameid-
         format:transient"
              name="urn:2.16.807.10.1.10.15"/>
</resolver:AttributeDefinition>
```

In this file, the section ref=”myLDAP” references which method is used for querying the attributes. Ref is set to myLDAP because we used our existing LDAP-proxy from before that is integrated with Microsoft AD, so all the attributes are queried from MS A the same placeD.

The configuration of myLDAP looks like this:

```
<resolver:DataConnector    id="myLDAP"    xsi:type="LDAPDirectory"
       xmlns="urn:mace:shibboleth:2.0:resolver:dc"
       ldapURL="ldap://10.200.10.132"
       baseDN="dc=yourdomain.com"
       principal="cn=admin,dc=yourdomain"
       principalCredential="password">
    <resolver:Dependency ref="krb_principalname" />
    <FilterTemplate>
       <![CDATA[
           (|(uid=$requestContext.principalName)
             (uid=${krb_principalname.get(0)}))
                 ]]>
    </FilterTemplate>

    <LDAPProperty         name="java.naming.ldap.attributes.binary"
       value="objectGUID"/>
</resolver:DataConnector>.
```

This does not mean that attributes can be kept only on LDAP, or in the Microsoft Active Directory. You could put here whatever database you want for storing all of your users and their data.

A small problem in this scenario could be the username. Students use student ID numbers as a username to log in to the faculty services, but all the users in Microsoft Office 365 must be in the format: **name@yourdomain.tld**

To solve this problem, you could make a small tweak in the attribute-resolver.xml, so when the user is successfully authenticated, and the IdP prepares to send the attributes to Office 365, instead of sending the student id number, it performs an LDAP query. When the user is found, it creates a UID in the following format: **surname.name@subdomain.yourdomain.com**. That is shown in the code above between the <FilterTemplate> tags. If your local usernames and email accounts are the same as in Office 365, this tweak will be unecessary.

Attribute filter

Every IdP should control which attributes are released to which service provider. That is possible by setting the **attribute-filter.xml in /opt/shibboleth-idp/conf**. For the Office 365 service we have set the following filter rules:

```
<!-- Release userPrincipalName as Windows Azure AD User ID -->
    <afp:AttributeFilterPolicy id="UserId">
        <afp:PolicyRequirementRule
       xsi:type="basic:AttributeRequesterString"
       value="urn:federation:MicrosoftOnline"/>
        <afp:AttributeRule attributeID="UserId">
            <afp:PermitValueRule xsi:type="basic:ANY"/>
        </afp:AttributeRule>
```

```
</afp:AttributeFilterPolicy>

<!-- Release Immutable ID to Windows Azure AD -->
<afp:AttributeFilterPolicy id="ImmutableID">
    <afp:PolicyRequirementRule
   xsi:type="basic:AttributeRequesterString"
   value="urn:federation:MicrosoftOnline"/>
    <afp:AttributeRule attributeID="ImmutableID">
        <afp:PermitValueRule xsi:type="basic:ANY"/>
    </afp:AttributeRule>
</afp:AttributeFilterPolicy>
```

2.2.2 Relaying party

This section of the configuration determines which party of the federation can use which service. Here the statement **id="urn:federation:MicrosoftOnline" provider=** says that the provider **yourdomain.com** can participate and exchange data with the MicrosoftOnline federation.

```
<!-- Windows Azure AD -->
    <rp:RelyingParty id="urn:federation:MicrosoftOnline"
              provider="https://yourdomain.com/idp/shibboleth"
              defaultSigningCredentialRef="IdPCredential">
        <rp:ProfileConfiguration xsi:type="saml:SAML2SSOProfile"
                                 signAssertions="conditional"
                                 encryptAssertions="never"
                                 encryptNameIds="never" />

        <rp:ProfileConfiguration xsi:type="ecp:SAML2ECP"
                                 includeAttributeStatement="true"
                                 assertionLifetime="300000"
                                 assertionProxyCount="0"
                                 signResponses="conditional"
                                 signAssertions="always"
                                 encryptAssertions="never"
                                 encryptNameIds="never"/>

        <rp:ProfileConfiguration xsi:type="saml:SAML2ECPProfile"
                                 includeAttributeStatement="true"
                                 assertionLifetime="PT5M"
                                 assertionProxyCount="0"
                                 signResponses="never"
      signAssertions="always"
                                 encryptAssertions="never"
                                 encryptNameIds="never"/>
```

```
    </rp:RelyingParty>
```

The next section specifies from where to search for the Microsoft Federation Metadata, when the IdP wants to communicate with that service.

```
<!-- Windows Azure AD Metadata -->
<metadata:MetadataProvider                                    id="OrgID"
      xsi:type="metadata:ResourceBackedMetadataProvider">
<metadata:MetadataResource xsi:type="resource:FilesystemResource"
       file="/opt/shibboleth-idp/metadata/WindowsAzureAD-
metadata.xml"/>
</metadata:MetadataProvider>
```

3 Office 365 domain federation

3.1 Enable your domain to be federated

After configuring the IDP, the next step is to make your desired domain available in Office 365 federated. This could be performed by using Windows Azure Active Directory Module for Windows PowerShell. This module, and instructions on its installation are given in the link below[1]. After installation, the following commands must be executed to complete the federation of your domain.

- Click on the Windows Azure Active Directory Module for Windows PowerShell Shortcut. Right Click and Run As Administrator.
- After this you will see an PowerShell console, and you need to set the credential variable by typing: $cred=Get-Credential
- After typing this command, a window is shown (Figure 3.1). Enter your Global Administrator account from Office 365.

Figure 3.1: Credential request screen

- Connect to Microsoft Online Services with the credential variable set previously: Connect-MsolService –Credential $cred

Then run the following commands to convert an existing domain (in this example, **subdomain.yourdomain.com**) for single sign on:

```
$dom = "subdomain.yourdomain.com”
$url = "https://yourdomain.com/idp/profile/SAML2/POST/SSO"
$ecpUrl = "https://yourdomain.com/idp/profile/SAML2/SOAP/ECP"
$uri = "https://yourdomain.com/idp/shibboleth"
$logouturl = "https://yourdomain.com/logout/"
$cert = "MIIFYzCCBEugAw...2tLRtyN"
```

[1] https://msdn.microsoft.com/en-us/library/azure/jj151815.aspx

```
Set-MsolDomainAuthentication –DomainName $dom -FederationBrandName
$dom -Authentication Federated -PassiveLogOnUri $url -
SigningCertificate $cert -IssuerUri $uri -ActiveLogOnUri $ecpUrl -
LogOffUri $logouturl -PreferredAuthenticationProtocol SAMLP
```

After running these commands, you can verify that your domain is federated by typing **Get-MsolDomain**. The output of this command shows all of the domains you have registered in Office 365, status and authentication. If there were no errors from the previous command the output should resemble:

```
PS C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Windows
Azure Active Directory> Get-MsolDomain

Name                          Status          Authentication
----                          ------          --------------
Subdomain.yourdomain.com      Verified        Federated
```

4 Sync tool

The last step in order to be able to log in to Office 365 is to create users that have the same attributes as the ones in the local database.

Microsoft suggest using the Microsoft tool called DyrSync. This tool is partially based on Microsoft Identity Life Cycle Management and enables the syncing of local Active Directory users and groups into the Azure Active Directory. This tool can be set up to automatically synchronise user accounts (creates, updates deletes) in Azure AD.

This tool is hard to control and was found to be complex and resource hungry in our experience. The tool produced unexpected behaviour, possibly due to its complex nature. It caused some of the users to get a bad domain and some user accounts were not created due to duplication of the users' UUID.

For better control of this process we discovered that there is a PowerShell library that implements all MS Online functionalities that DirSync tool makes. The following examples are code snippets from our own dirsync tool.

Importing of the module library:

```
import-module MSOnline
```

Creating new user accounts that are in a csv format:

```
import-csv createMSOL.csv | New-MsolUser
```

Updating user accounts:

```
import-csv updateMSOL.csv | Set-MsolUser
```

Applying licences to user accounts:

```
import-csv createMSOL.csv | select UserPrincipalName | Set-
      MsolUserLicense                                -AddLicenses
      "yourdomain.com:STANDARDWOFFPACK_STUDENT"
```

The following is the format of the Comma Separated Values (.CSV) file that is used for user information:

```
""UserPrincipalName,""ImmutableId"",""FirstName"",""LastName"",
""DisplayName"",""UsageLocation""
```

The main key that is used to link a logged-in user that is sent using the SAML2 message to Office 365 is the **ImmutableID** i.e., the UUID of the user.

We have developed a tool that integrates and automates this process using these scripts. It can take information from different data sources like Microsoft AD, LDAP, Database, etc.

5 Conclusion

By federating students' domain with Office 365, the management of users is much easier and much quicker.

Now it takes only a few minutes to create email accounts for new students. Also, there is no need for students to remember different username and passwords for all of the services that they are using. With a single login they access all of the services that are available to them and the problem with resetting forgotten passwords is solved. The benefit from this integration was felt both from the students and the Administrators. This federation provides new services from Microsoft that are easily integrated and used by the students, providing a great opportunity to get hands-on experience with "real world" software and services that are usually expensive.

References

[Shibboleth] https://shibboleth.net/

[MSOffice 365] http://office.microsoft.com/en-001/products/?CTT=97

[DomFed] http://blogs.technet.com/b/canitpro/archive/2013/06/13/step-by-step-setting-up-ad-fs-and-enabling-single-sign-on-to-office-365.aspx

[Shibboleth] Set up a trust between Shibboleth and AzureAD
http://technet.microsoft.com/en-us/library/jj205457.aspx

[ShibbolethSSO] Configure Shibboleth for use with single sign-on
http://technet.microsoft.com/en-us/library/jj205463.aspx

Glossary

AD	Active Directory
CAS	Central Authentication Service
DNS	Domain Name Server
IdP	Identity Provider
JASIG	Java in Administration Special Interest Group, a non-profit US organisation
LDAP	Lightweight Directory Access Protocol
MS	Microsoft (Corporation)
ASOffice 365	The brand name used by Microsoft for a group of software plus services subscriptions that provides productivity software (Microsoft Office) and related services (OneDrive, Skype, etc.) to its subscribers**.**
SAML2	Security Assertion Markup Language v2.0
SSO	Single Sign-On
UUID	Unique User ID

Complete BPDs are available at http://services.geant.net/cbp/Pages/Home.aspx

campus-bp-announcements@terena.org