Membangkitkan XACML Request Menggunakan Framework X-CREATE

Tugas Akhir Mata Kuliah Keamanan Perangkat Lunak EL5215

Silvana Rasio Henim
23215068

Magister Teknik Elektro
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2016

ABSTRAK

Mekanisme keamanan pada pengaturan *web service* melindungi kerahasiaan dan integritas informasi. Kebijakan pengontrolan akses (*access control policy*) merupakan salah satu standar keamanan pada *web service*. Kebijakan tersebut mengatur berbagai hal, seperti tingkat kerahasiaan data, pengaturan data dan sumber daya, melakukan klasifikasi terhadap data dan sumber daya menjadi beberapa kategori dengan kontrol akses yang berbeda. Penerapan *access control policy* dapat dilakukan dengan menggunakan eXtensible Access Control Markup Language (XACML). Pendekatan yang paling umum digunakan untuk menguji XACML *policy* adalah menggunakan XACML *request*. XACML *request* akan digunakan sebagai masukan dari PDP. XACML *request* dapat ditulis secara manual, akan tetapi membutuhkan usaha yang besar. Untuk itu dibutuhkan sebuah *tool* yang dapat menghasilkan XACML *request* secara otomatis. Salah satu *tool* yang dapat digunakan adalah X-CREATE. Berdasarkan pengujian dengan menggunakan beberapa masukkan berupa *file* XACML *policy*, X-CREATE dapat menghasilkan XACML *request.*

Kata kunci : keamanan, web service, access control policy, XACML request, X-CREATE

I. PENDAHULUAN

Web service mempertukarkan data dalam format XML *message* melalui jaringan, menggunakan protokol HTTP. Mekanisme keamanan pada pengaturan *web service* melindungi

kerahasiaan dan integritas informasi. Informasi yang dimaksud adalah data yang dipertukarkan antara *client* dan *server*. Standar keamanan pada *web service* dibuat karena kebutuhan penyediaan keamanan pada *web service* tersebut, salah satunya adalah dengan kebijakan pengontrolan akses (*access control policy*). Kebijakan tersebut mengatur berbagai hal, seperti tingkat kerahasiaan data, pengaturan data dan sumber daya, melakukan klasifikasi terhadap data dan sumber daya menjadi beberapa kategori dengan kontrol akses yang berbeda. Pada pengaturan *web service, access controls policies* diimplementasikan dengan melakukan konfigurasi pada setiap *node* sehingga membuat perubahan kebijakan menjadi sangat mahal dan tidak handal. *Access control policies* sering dibuat menggunakan bahasa yang berbeda-beda dan *proprietary languanges*, sehingga mengakibatkan kebijakan tersebut sulit untuk digunakan oleh aplikasi lain (berbagi pakai). Salah satu solusi permasalahan tersebut adalah eXtensible Access Control Markup Language (XACML) yang merupakan bahasa berbasis XML yang distandarisasi oleh *Organization for the Advancement of Structured Information Standards* (OASIS). XACML memberikan sebuah model dan bahasa untuk mengekspresikan kebijakan pengontrolan akses yang dapat diterapkan untuk layanan web serta sumber daya lainnya. Selain XACML OASIS juga mendefenisikan arsitektur dan komponen untuk membuat dan mengatur kebijakan pengontrolan akses yang ditulis dalam XACML[5].

Salah satu komponen arsitektur yang didefenisikan oleh OASIS adalah *Policy Decision Point* (PDP). PDP merupakan komponen penting dalam pengambilan keputusan akses yang berpotensi untuk dipanggil ketika ada permintaan (*request*). PDP mengevaluasi permintaan pengontrolan akses yang diterimanya sebagai masukan terhadap sekumpulan kebijakan XACML dan kemudian mengembalikan keputusan seperti diterima atau ditolak[5].

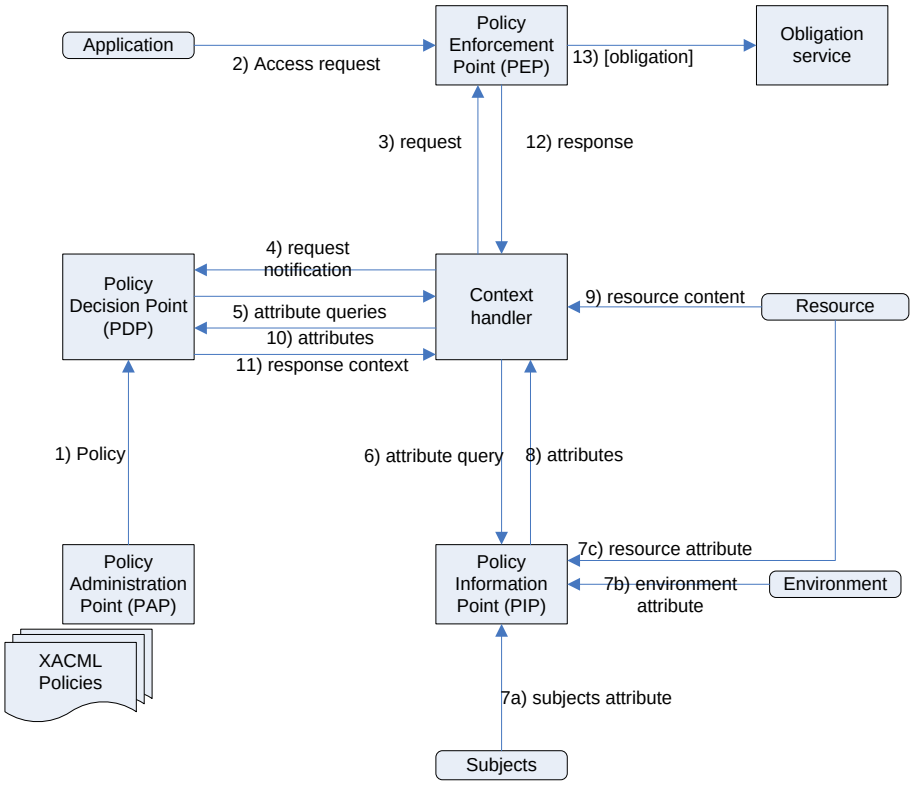
Access control policies harus dirancang serta diuji dengan hati-hati untuk melindungi data dari akses yang tidak sah. Pendekatan yang umum digunakan untuk menguji policy XACML adalah dengan derivasi input tes (XACML *request*) yang digunakan untuk menyelidiki PDP dan memeriksa PDP *response* dibandingkan dengan yang diharapkan. XACML sangat kompleks sehingga membutuhkan usaha yang besar untuk menghasilkan XACML *request* apabila dilakukan secara manual. Salah satu *tool* yang dapat digunakan untuk menghasilkan XACML *request* adalah X-CREATE (XaCml REquests derivAtion for TEsting). X-CREATE merupakan sebuah *framework* yang dikembangkan oleh Antonia Bertolino, dkk.

Pada tulisan ini membahas hal-hal berikut: pada bagian II membahas XACML, bagian III membahas framework X-CREATE serta metode yang digunakan untuk menghasilkan XACML

request, bagian IV membahas langkah-langkah untuk menghasilkan XACML *request* menggunakan X-CREATE dan bagian terakhir berisi kesimpulan.

II. eXTENSIBLE ACCESS CONTROL MARKUP LANGUAGE (XACML)

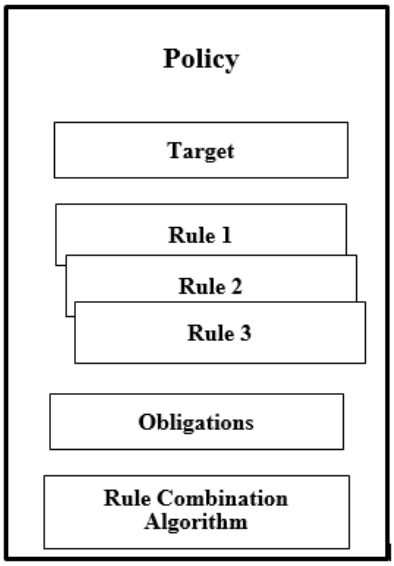
XACML merupakan standar dari OASIS yang menjelaskan mengenai *policy language* dan *access control decision request/ response language* yang keduanya ditulis dalam format XML. *Policy language* digunakan untuk menjelaskan kebutuhan pengontrolan akses secara umum dan memiliki *standard extension points* untuk mendefenisikan fungsi baru, tipe data, kombinasi logika, dan lain-lain. *Request/response language* digunakan untuk membentuk sebuah *query* untuk mengevaluasi apakah suatu tindakan diizinkan untuk dilakukan ataukah tidak dan untuk menginterpretasikan hasilnya[7]. XACML versi 2.0 telah distandarisasi oleh OASIS pada February 2005.



Gambar 1. XACML Data Flow Model [1]

XACML *data flow model* menggambarkan logika yang terlibat dalam melakukan pemrosesan terhadap sebuah permintaan akses. Seperti yang terlihat pada Gambar 1 XACML *context handler* memisahkan aplikasi dari *canonical representation* untuk masukan dan keluaran

yang digunakan oleh PDP. XACML mendefenisikan *syntax* untuk *policy languages*, semantik untuk memproses kebijakan dan protokol *request-response* antara PEP dan PDP.



Gambar 2 XACML Policy Elemen [6]

XACML Policy terdiri dari tiga bagian seperti yang terlihat pada Gambar 2, yaitu:

a. *Policy Target*

Policy Target dapat terdiri *Subject, Resource* dan *Action* dengan jumlah yang berbeda untuk setiap unsur didalamnya. *Subject* merepesentasikan identitas dari entitas yang melakukasn aksi. *Resource* merepresentasikan sumber daya yang akan diakses oleh pengguna. *Action* menggambarkan aktivitas yang dilakukan oleh pengguna pada sebuah *resource* (seperti: membaca, menulis, mengeksekusi, dan lain-lain)

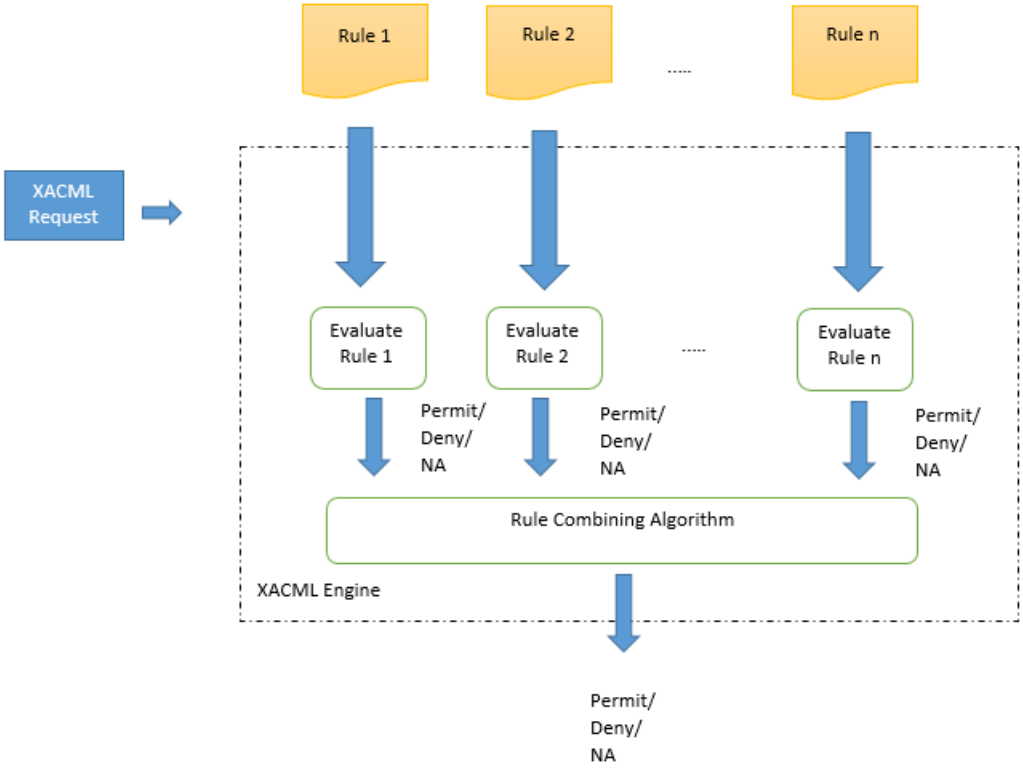
b. *Rules*

Merupakan bagian utama dari XACML. *Rules* berisi ekspresi logika yang harus dipenuhi oleh sebuah *request* agar *rule* dapat diterapkan. Sama halnya dengan Policy target, *rules* juga dapat terdiri *Subject*, *Resource* dan *Action* untuk mencocokkan antara aturan dan *request*. *Rules* diterapkan untuk menentukan apakah *request* yang ada "*permit/deny*" ataukan "*not applicable*" untuk melakukan akses.

c. *Obligation*

Obligation digunakan untuk memanggil aksi tertentu selama evaluasi Policy dilakukan. Misalkan jika hasil evaluasi *policy* adalah "izinkan", maka aksi yang dilakukan adalah mengirimkan email ke kepala departemen[5].

Selain tiga komponen diatas, sebuah *policy* juga memiliki *Rule combining algorithm*. *Rule combining algorithm* menentukan pendekatan yang digunakan untuk mengkombinasikan keputusan ketika lebih dari satu *rule* yang diterapkan pada satu *request* (Gambar 3). Contohnya *permit-override*, ketika melakukan evaluasi terhadap *policy* jika minimal satu *policy* menghasilkan *"permit"*, maka keluarannya adalah *"permit"*[]. Jika terjadi kesalahan pada saat menerapkan *policy* pada sebuah *request*, maka akan mengembalikan *Indeterminate* sebagai sebuah *decision*.



Gambar 3 *Rule Combining Algorithm* [6]

Contoh dari XACML Policy dapat dilihat pada Listing 1.

```
<Policy xmlns= " urn : oasis : names : tc : xacml : 1.0 : policy "
 PolicyId= " Policy Example "
 RuleCombiningAlgId= " rule-combining- algorithm : first-applicable "
>
  <Target>
    <Subjects><AnySubject/></Subjects>
    <Resources><AnyResource/></Resources>
    <Actions><AnyAction/></Actions>
  </Target>
  <Rule RuleId= " PolicyExample : rule1 " Effect= " Deny " >
    <Target>
     <Subjects><AnySubject/></Subjects>
   <Resources>
     <Resource>
        <ResourceMatch
```

```
        MatchId= " a n y U R I - e q u a l " >
          <AttributeValue  DataType= " a n y U R I "
>http://library.com/record</AttributeValue>
          <ResourceAttributeDesignator       AttributeId= " r e s o u r
c e - i d " DataType= " a n y U R I " />
        </ResourceMatch>
      </Resource>
     </Resources>
    <Actions>
      <Action>
        <ActionMatch
        MatchId= " s t r i n g - e q u a l " >
          <AttributeValue
          DataType= " s t r i n g " >write</AttributeValue>
          <ActionAttributeDesignator
          AttributeId= " a c t i o n - i d " DataType= " s t r i n g "
/>
        </ActionMatch>
      </Action>
     </Actions>
    </Target>
    <Condition
     FunctionId= " f u n c t i o n : n o t " >
       <Apply
       FunctionId= " string - at - least - one - member -o f " >
       <SubjectAttributeDesignator AttributeId= " r o l e "
MustBePresent= " f a l s e " DataType= "s t r i n g " />
        <Apply FunctionId= " s t r i n g - b a g " >
          <AttributeValue
          DataType= " s t r i n g " >researcher</AttributeValue>
          <AttributeValue
          DataType= " s t r i n g " >professor</AttributeValue>
          <AttributeValue
          DataType= " s t r i n g " >staff</AttributeValue>
        </Apply>
      </Apply>
     </Condition>
   </Rule>
   <Rule
   RuleId= " P o l i c y E x a m p l e : r u l e 2 " Effect= " P e r m
i t " >
   <Target>
      <Subjects>
        <Subject>
          <SubjectMatch
          MatchId= " s t r i n g - e q u a l " >
          <AttributeValue
          DataType= " string " >Julius</AttributeValue>
          <SubjectAttributeDesignator
          AttributeId= " subject - i d " DataType= " string" />
          </SubjectMatch>
        </Subject>
      </Subjects>
```

```
            <Resources>
              <Resource>
                <ResourceMatch
                MatchId= " anyURI - equal " >
                <AttributeValue
                DataType= " anyURI " >http://library.com/record/
               journals</AttributeValue>
                <ResourceAttributeDesignator
                AttributeId= " resource - id " DataType= " anyURI " />
                </ResourceMatch>
              </Resource>
            </Resources>
            <Actions>
              <Action>
                <ActionMatch
                MatchId= " s t r i n g - e q u a l " >
                <AttributeValue
                DataType= " s t r i n g " >read</AttributeValue>
                <ActionAttributeDesignator
                AttributeId= " action - i d " DataType= " string " />
                </ActionMatch>
              </Action>
            </Actions>
          </Target>
        </Rule>
</Policy>
```

Listing 1 Contoh XACML Policy [3]

Pendekatan yang paling umum digunakan untuk menguji XACML *policy* adalah menggunakan XACML *request*. XACML *request* akan digunakan sebagai masukan dari PDP. Respon dari PDP akan dibandingkan dengan respon yang diharapkan. XACML *request* dapat ditulis secara manual dalam format xml, akan tetapi mengingat kompleksitas dari XACML *request* maka membutuhkan usaha yang besar apabila dilakukan secara manual. Salah satu *tool* yang dapat digunakan untuk menghasilkan XACML *request* dari sebuah XACML *Policy* adalah X-CREATE.

III. FRAMEWORK X-CREATE

X-CREATE (Xacml Requests derivation for Testing) merupakan sebuah *tool* untuk membangkitkan *test case* XACML *policy*. Untuk menghasilkan XACML *request* X-CREATE menggunakan tiga strategi, yaitu:

a. Simple combinatorial strategy

Strategi ini menerapkan sebuah pendekatan kombinasi sederhana pada *policy values*. Pada strategi ini didefenisikan himpunan SubjectSet, ResourceSet, ActionSet dan EnvironmentSet serta mempertimbangkan entitas acak untuk ketahanan. Secara khusus entitas *subject* didefenisikan

sebagai kombinasi dari nilai-nilai elemen dan atribut himpunan SubjectSet. Begitu pula dengan entitas *resource*, entitas *action* dan entitas *environment* yang didefenisikan sebagai kombinasi dari nilai-nilai elemen dan atribut himpunan ResourceSet, ActionSet dan EnvironmentSet.

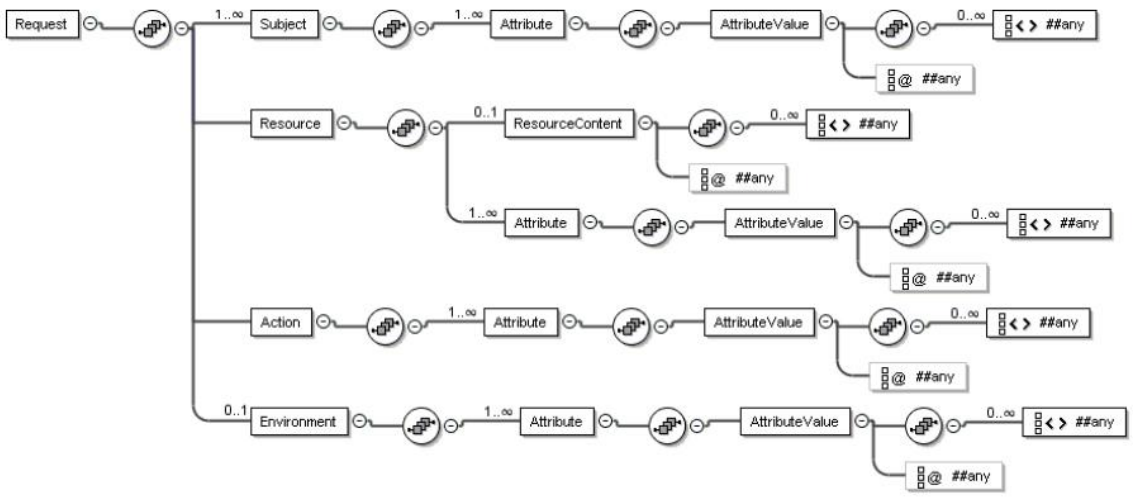
Langkah selanjutnya adalah menghasilkan seluruh kombinasi dari entitas *subject*, entitas *resource*, entitas *action* dan entitas *environment* yang terdapat di dalam himpunan dengan cara sebagai berikut:

1. Menerapkan kombinasi berpasangan sehingga diperoleh PW *set*.
2. Menerapkan *three-wise combination* sehingga diperoleh TW *set*.
3. Menerapkan *four-wise combination* sehingga diperoleh FW *set*.

Inklusi dari *set* ini dapat ditulis sebagai $PW \subseteq TW \subseteq FW$. Untuk menghilangkan kombinasi yang berulang (duplikat) dilakukan dengan mempertimbangkan kombinasi : PW disebut dengan *Pairwise*, $TW \backslash PW$ disebut *Threewise*, dan $FW(TW \cup PW)$ disebut *Fourwise.* Jumlah maksimum XACML *request* yang dihasilkan dengan strategi ini sama dengan kardinalitas dari FW *set.* Angka ini menunjukkan *stopping criteria* untuk *simple combinatorial strategy*. Keuntungan dari strategi ini adalah sederhana dan dapat mencapai cakupan domain masukan dari *policy* yang direpresentasikan oleh *policy values combination* [3].

b. XPT strategy

Pada strategi ini pendekatan *XML-based Partition Testing* (XPT) digunakan untuk menghasilkan XACML *request* dari XACML *Context Schema*. XACML *Context Schema* untuk menghasilkan XACML *request* dapat dilihat pada Gambar 4.



Gambar 4 XACML *Context Schema* [2]

XPT strategy ini memiliki tiga langkah utama, yaitu:

1. *Intermediate request generation*

Pada langkah ini XML *instance* dihasilkan dari XACML *schema* dengan menerapkan sebuah varian metode *Category Partition* (CP) dan *traditional boundary condition*. Secara khusus dalam menghasilkan XML *instance* angka kejadian untuk setiap elemen di dalam skema dianalisa dan menerapkan *boundary condition* serta mempertimbangkan nilai-nilai perbatasan (minOccurs dan maxOccurs). Listing 2 merupakan contoh dari *intermediate generated request* dimana nilai dari setiap elemen dan atribut belum ditentukan.

```
<Request xmlns= "urn : oasis : names : tc : xacml : 1.0 : context " >
  <Subject>
    <Attribute AttributeId= " " DataType= " " >
      <AttributeValue/>
    </Attribute>
  </Subject>
  <Resource>
    <Attribute AttributeId= " " DataType= " " >
      <AttributeValue/>
    </Attribute>
    <Attribute AttributeId= " " DataType= " " >
      <AttributeValue/>
    </Attribute>
  </Resource>
  <Action>
    <Attribute AttributeId= " " DataType= " " >
    <AttributeValue/>
    </Attribute>
  </Action>
</Request>
```

Listing 2 Contoh *Intermediate request* [2]

2. *Policy-under-test analisys*

Nilai elemen dan atribut pada *intermediate request* diambil dari nilai hasil *policy under test* sehingga menghasilkan *final request* yang dapat dieksekusi. Langkah-langkah yang dilakukan pada tahapan ini adalah:

- Mendefenisikan empat himpunan nilai: SubjectSet, ResourceSet, ActionSet dan EnvironmentSet

- Mencari elemen dan atribut yang cocok, yang terdapat pada elemen target untuk setiap elemen *rule*, *policy* dan *policySet* yang termasuk dalam *policy under test*. Elemen dan atribut ini didefenisikan dalam bagian SubjectMatch, ResourceMatch dan ActionMatch dari *subjects*, *resources* dan *actions*.
- Mengambil nilai dari elemen AttributeValue dan nilai dari AttributeId, Datatype, Issuer dan SubjectCategory dari SubjectMatch, ResourceMatch dan ActionMatch serta meletakkannya pada SubjectSet, ResourceSet, ActionSet
- Untuk setiap *policy rule*, mengambil seluruh nilai dari elemen AttributeValue dan nilai dari AttributeId, Datatype dan Issuer (opsional) dari bagian *condition* serta meletakkannya pada SubjectSet, ResourceSet, ActionSet dan EnvironmentSet, jika merujuk pada elemen SubjectAttributeDesignator, ResourceAttributeDesignator, ActionAttributeDesignator or EnvironmentAttributeDesignator.

Hasil dari *policy-under-test analisys* untuk Listing 1 dapat dilihat pada Tabel 1 berikut.

Tabel 1 Hasil *policy-under-test analisys* [2]

AttributeId	**DataType**	**AttributeValue**
SubjectSet		
role	string	professor
		researcher
		staff
subject-id	string	Julius
ResourceSet		
resource-id	anyURI	http://library.com/record
resource-id	anyURI	http://library.com/record/journals
ActionSet		
action-id	string	read
action-id	string	write

3. *Request value assignment*.

Tahapan ini dilakukan pengisian *intermediate request* dengan nilai dari SubjectSet, ResourceSet, ActionSet dan EnvironmentSet [2]. Listing 3 merupakan *final request* yang diperoleh dengan mengisi nilai *intermediate request* dengan hasil analisa Tabel 1

```
<Request xmlns= "urn : oasis : names : tc : xacml : 1.0 : context " >
  <Subject>
    <Attribute AttributeId= "subject-id " DataType= "string " >
      <AttributeValue> Julius </AttributeValue>
    </Attribute>
  </Subject>
  <Resource>
    <Attribute AttributeId= "resource-id " DataType= "anyURI " >

  <AttributeValue>http://library.com/record/journals</AttributeValue>
    </Attribute>
    <Attribute AttributeId= "datastream:id " DataType= ""anyURI " >
      <AttributeValue>http://library.com/record</AttributeValue>
    </Attribute>
  </Resource>
  <Action>
    <Attribute AttributeId= "action-id " DataType= "string " >
    <AttributeValue>read</AttributeValue>
    </Attribute>
  </Action>
</Request>
```

Listing 3 Final request

c. Multiple combinatorial strategy

Multiple combinatorial strategy menggunakan pendekatan *combinatorial*. Pada strategi ini untuk setiap *policy*, dihasilkan empat set, yaitu SubjectSet, ResourceSet, ActionSet dan EnvironmentSet yang berisi nilai dari elemen dan atribut *subject, resource, action* dan *environment*. Sebuah entitas subject merupakan kombinasi nilai dari atribut `< AttributeId>` dan `<DataType>` serta nilai dari `<AttributeValue>`. Hal yang sama juga berlaku untuk entitas resource, action dan environment[4].

IV. MEMBANGKITKAN XACML REQUEST DENGAN X-CREATE

X-CREATE dapat diunduh pada halaman *website* http://labsedc-wiki.isti.cnr.it/labsedc/tools/xcreate/public/main#publications. Selain X-CREATE perangkat lunak yang dibutuhkan adalah *database engine* MySQL. Pada MySQL buat sebuah *database*

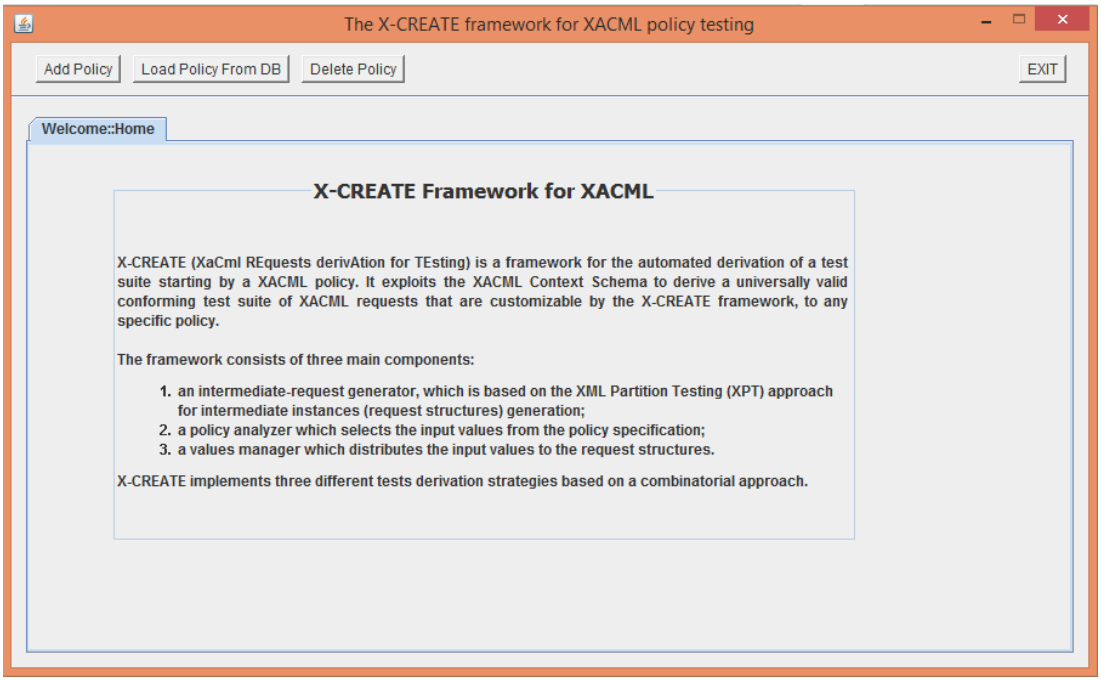
misalkan *database* xcreate. Kemudian X-CREATE yang telah diunduh (dalam format .zip) diekstrak. Supaya dapat dijalankan, terlebih dahulu dilakukan pengaturan properti X-CREATE yang terdapat pada *file* xcreate.properties. Isi dari *file* xcreate.properties dapat dilihat pada Listing 4 berikut:

```
#*******************************************************
# xcreate properties
#*******************************************************

#*******************************************************
#1. Modify for your database:
#*******************************************************
#MySQL
jdbc.url=jdbc:mysql://localhost:3306/xcreate


#*******************************************************
#2. Enter the username with which you connect to your database:
#*******************************************************
jdbc.username=silvana

#*******************************************************
#3. Enter the password with which you connect to your database:
#*******************************************************
jdbc.password=silvana84

#*************************************************************
#8. Specify if the user want use automatic statistical or not (true for
using the following values, false otherwise)
#*************************************************************
#statistic.bool=True
statistic.bool=False
valore.minEnv=1
valore.minSub=1
valore.minRes=1
valore.minAct=1
valore.maxSub=3
valore.maxRes=3
valore.maxAct=3
valore.maxEnv=3
```

Listing 4 file xcreate.properties

Bagian yang harus dilengkapi pada xcreate.properties adalah `jdbc.url`, `jdbc.username` dan `jdbc.password`. X-CREATE dapat dijalankan dengan mengetikkan perintah `java -jar xcreate.jar` pada *command prompt* (pastikan berada pada *home directory* xcreate), atau dengan melakukan klik ganda pada file xcreate.jar. Ketika pertama kali dijalankan X-CREATE secara otomatis akan membuat folder Temp_X-CREATE yang nantinya akan digunakan sebagai tempat penyimpanan XACML *request* yang dihasilkan. Selain itu X-

CREATE juga secara otomatis akan membuat tabel-tabel yang dibutuhkan di dalam *database*. Tampilan awal dari X-CREATE dapat dilihat pada Gambar 4 berikut:



Gambar 4 Tampilan awal X-CREATE

File access policy dapat ditambahkan dengan mengklik tombol Add Policy, kemudian memilih *file access policy* yang akan ditambahkan. Sebagai contoh adalah *file* policy_27.xml yang dapat diunduh di http://wso2.com/files/policy_27.xml. Isi dari *file* policy_27.xml seperti pada Listing 5 sebagai berikut:

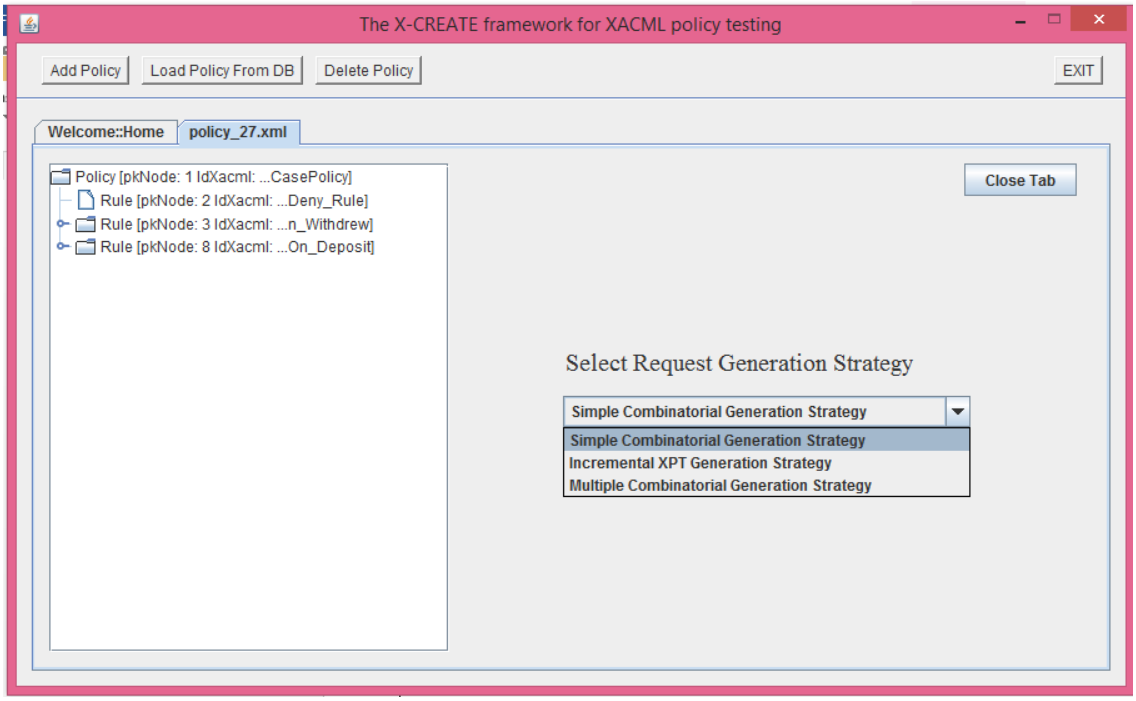
```
<?xml version="1.0" encoding="UTF-8"?>
<Policy PolicyId="BankUseCasePolicy"
RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-
algorithm:permit-overrides"
xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os">

<Target>
    <Environments>
      <Environment>
        <EnvironmentMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">wso2.org</AttributeValue>
          <EnvironmentAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:environment:environment-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
        </EnvironmentMatch>
      </Environment>
    </Environments>
  </Target>
  <Rule Effect="Permit" RuleId="Rule_On_Withdrew">
    <Condition>
```

```
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:and">
        <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-is-
in">
          <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">BankService/withdrew</Att
ributeValue>
          <ResourceAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
        </Apply>
        <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-is-
in">
          <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">manager</AttributeValue>
          <SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
        </Apply>
      </Apply>
    </Condition>
  </Rule>
  <Rule Effect="Permit" RuleId="Rule_On_Deposit">
    <Condition>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:and">
        <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-is-
in">
          <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">BankService/deposit</Attr
ibuteValue>
          <ResourceAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
        </Apply>
        <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:or">
          <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-
is-in">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">manager</AttributeValue>
            <SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
          </Apply>
          <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-
is-in">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">executive</AttributeValue
>
            <SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
          </Apply>
        </Apply>
      </Apply>
    </Condition>
  </Rule>
  <Rule Effect="Deny" RuleId="Deny_Rule"/>
```

```
</Policy>
```

Listing 5 policy_27 . xml

Tampilan setelah *file policy* ditambahkan dapat dilihat pada Gambar 5 berikut:



Gambar 5 Tampilan setelah penambahan *file policy*

XACML *request* yang dihasilkan oleh X-CREATE dengan menggunakan *simple combinatorial generation strategy* dapat dilihat pada Listing 6 berikut ini.

```
<?xml version="1.0" encoding="UTF-8"?>
<Request xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:context:schema:os
access_control-xacml-2.0-context-schema-os.xsd">
    <Subject>
      <Attribute AttributeId="http://wso2.org/claims/role"
DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>RandomValue:[B@f783b3c:3
        </AttributeValue>
      </Attribute>
    </Subject>
    <Resource>
      <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-
id" DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>http://localhost:9766/services/RestService/POST
        </AttributeValue>
      </Attribute>
    </Resource>
    <Action>
      <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>DELETE
```

```
        </AttributeValue>
      </Attribute>
    </Action>
    <Environment>
    </Environment>
</Request>
```

Listing 6 XACML Request

Hasil ini tersimpan di dalam folder `Temp_X-CREATE/nama_policy/nama_strategy`, contoh: `Temp_X-CREATE/policy_27/simple comb`. Pengujian dilakukan dengan menggunakan *file* beberapa XACML *policy* serta menggunakan pendekatan *simple combinatorial strategy, XPT strategy* dan *multiple combinatorial strategy*.

V. KESIMPULAN

Salah satu metode untuk mengamankan informasi yang terdapat pada *web service* adalah dengan melakukan pengontrolan akses menggunakan XACML *policy*. Pengujian XACML *policy* dapat dilakukan dengan memeriksa respon yang diberikan oleh PDP. Masukkan yang diberikan kepada PDP adalah XACML *request*. XACML *request* dapat dibuat secara otomatis dengan menggunakan X-CREATE. Untuk menghasilkan XACML *request*, X-CREATE menggunakan tiga pendekatan, yaitu *simple combinatorial strategy*, XPT *strategy* dan *multiple combinatorial strategy*.

REFERENSI

[1] Bertino, E., Martino, L., Paci, F., Squicciarini, A., Security for Web Services and Service-Oriented Architetures, Springer, New York, 2009.

[2] Bertolino, A., Lonetti, F., Marchetti, E., Systematic XACML request generation for testing purposes, *IEEE computer society,* pp. 3-11, 2010.

[3] Bertolino, A., Daoudagh, S., Lonetti, F., Marchetti, E., The X-CREATE Framework-A Comparison of XACML Policy Testing Strategies, *WEBIST,* pp. 155-160, 2012.

[4] Bertolino, A., Daoudagh, S., Lonetti, F., Marchetti, E., Schilders, L., Automated testing of eXtensible Access Control Markup Language-based access control systems, *IET Software,* pp. 203-212, **7**, 2013.

[5] Turkmen, F., Crispo, B., Performance Evaluation of XACML PDP Implementations, *Proceedings of the 2008 ACM workshop on Secure web services*, pp 37-44, 2008.

[6] Jayasekara, A., http://wso2.com/library/articles/2011/10/understanding-xacml-policy-language-xacml-extended-assertion-markup-langue-part-1/ diakses pada 20 Mei 2016.

[7] _____., https://www.oasis-open.org/committees/download.php/2713/Brief_Introduction_to_XACML.html diakses pada 20 Mei 2016