BAB 2
Tinjauan Pustaka

Access control memiliki beberapa macam model yang digunakan mulai dari pertama kali sampai dengan yang terbaru. Adapun model dari *access control* yaitu antara lain Mandatory Aceess Control (MAC), Discretionary Access Control (DAC) , Roll Based Access Control (RBAC), dan Attribute Based Access Control (ABAC):

1. MAC (Mandatory Access Control)

   Model MAC merupakan model *access control* dimana sistem yang akan memutuskan bagaimana data akan diakses. Pemberian akses bergantung pada pemilik *document.*
2. DAC (Discretionary Access Control)

   DAC merupakan model *access control* dimana *user* yang memutuskan bagaiman *user* melindungi datanya, melalui sistem komputer untuk untuk membatasi akses ke objek berdasarkan identitas subjeknya.
3. RBAC (Roll Based Access Control)

   RBAC merupakan perpaduan antara MAC dan DAC dimana pendekatan yang membatasi akses sebuah sistem untuk *user* yang mempunyai kewenangan dalam sistem tersebut.
4. ABAC (Attribute Based Access Control)

   Salah satu model *access control* yang menerapkan policy didalamnya.

Menurut Xu & Zhang (2014), generasi terbaru dari model *access control* yaitu model ABAC, model ABAC ini memiliki beberapa fitur, antara lain:

1. ABAC dapat melakukan pemberian grant *access control* melalui *attribute* dari elemen-elemen authorisasi seperti subjek, resource, action dan environment menjadi satu keputusan *access control.* Hal ini juga memungkinkan subjek untuk mengakses resource seluas mungkin tanpa adanya hubungan individual antara setiap subjek.
2. ABAC dapat memfasilitasi *collaboration policy adminnistration*dalam organisasi yang besar. Policy tersebut dapat disusun oleh pembuat policy yang berasal dari department yang berbeda.
3. ABAC juga dapat memfasilitasi proses *decoupling access control* dari logika bisnis sebuah aplikasi tertentu.

ABAC merupakan metode *access control* dimana subjek hanya akan melakukan *request* untuk menjalankan operasi terhadap objek yang didasarkan pada *attribute* yang telah disematkan pada subjek, objek, kondisi lingkungan dan beberapa policy yang termasuk dalam *attribute* dan kondisinya tersebut. Pada model ABAC elemen *authorisasi* didefinisikan di dalam *attribute.*

Menurut Sandhu (2010) terdapat 4 aspek *attribute* dalam ABAC yaitu:

1. *Subject*

   Subject merupakan pengguna, baik manusia mapun bukan (misal device atau *software*) yang meminta *request access.* Contoh dari *subject* tersebut antara lain nama, alamat, posisi jabatan, dan lain-lain. Sedangkan *request* dapat menggunakan *attribute* dari *subject* tersebut dengan bersifat unik.

2. *Resource*

   Resource merupakan target yang diproteksi misalnya device, jaringan, file, aplikasi, dan lain sebagainya.

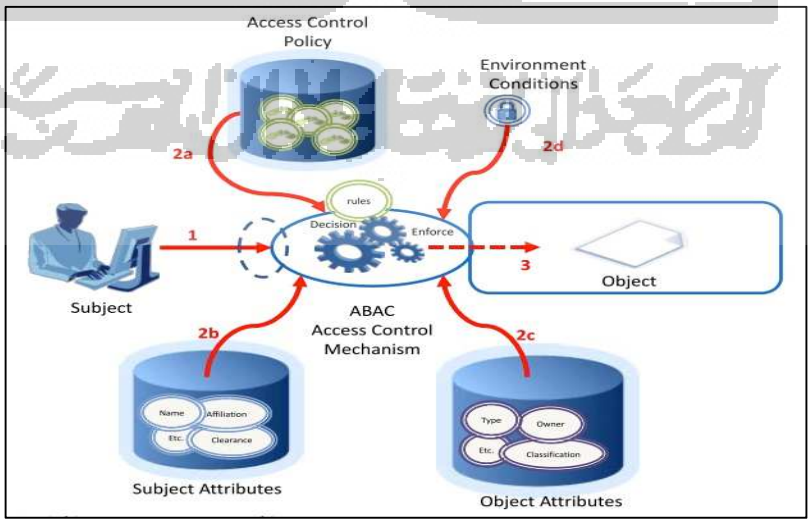
3. *Operation*

   Operation merupakan eksekusi dari suatu fungsi yang sedang melakukan *request* dari sebuah subjek terhadap *resource*.

4. *Environment attribute*

   Merupakan karakteristik dari operasional maupun situasional, contohnya seperti *current time*, *ip address* dan lain sebagainya.

Cara kerja dari ABAC menurut Hu et al. (2014) dapat digambarkan pada gambar 2.1:



Gambar 2.1 Gambaran umum cara kerja ABAC (Hu et al. 2014)

Gambar tersebut dapat dijelaskan bahwa terdapat 3 langkah utama dalam menerapkan ABAC yaitu :

1. *Subject* menerima *request* dari *object*
2. Pemberian *decision* diberikan melalui mekanisme evaluasi terhadap (a) *Rules*, (b) *Subject Attribute*, (c) *Object Attribute* dan (d) kondisi *environment*.
3. *Subject* diberikan *decision* : *deny* atau *permit* untuk akses terhadap *object*.

Sebuah *attribute* dapat dispesifikasikan melalui sebuah *identifier* (variabel), tipe data dan sebuah domain dimana sebuah himpunan finite yang memuat nilai tipe data yang diberikan. Tipe data dari *attribute* dapat berupa tipe data yang umumnya dipakai dalam sistem komputer seperti: *integer*, *string* dan *boolean*. Tipe data atau domain dari *attribute* pada ABAC dapat dispesifikasikan secara eksplisit ataupun implisit.

Sebuah *policy* ABAC merupakan representasi dari fungsi yang menentukan apakah permintaan akses dibolehkan berdasarkan nilai *attribute* yang diberikan. Secara formal sebuah *policy* ABAC akan memuat triple (X,Y,f). Dimana

- X adalah himpunan finite dari attribute dengan domain D1…Dn
- Y adalah himpunan finite dari *access control decision* (misalnya : *permit, deny, undefined*)
- f := D1 x D2 x … Dn → Y ; inilah fungsi *access control*

Sebuah *policy* ABAC dikatakan sebagai *complete* bila dan hanya bila f adalah fungsi total, dimana untuk nilai yang diberikan dari setiap *attribute* maka f selalu menghasilkan sebuah *deterministic decision*. Dalam hal ini sistem ABAC yang berbeda akan menggunakan himpunan keputusan *access control* yang berbeda, misalnya: {*permit, deny, undefined*} atau {*permit, deny, NotApplicable, Intermediate*}.

Menurut Aqib & Shaikh (2015) terdapat 2 masalah yang dihadapi dalam menerapkan solusi *access control* yaitu:

1. *Inconsistency*

Inconsistency yaitu kondisi dimana terdapat 2 *rule* yang memberikan hasil kontradiksi. Bila S, O dan A adalah *Subject*, *Object* dan *Actions*. Bila diberikan $a \in$ A, $s \in S$, $o \in O$, kemudian diberikan $d \in D$ yaitu himpunan *Decision* D = { permitted,

denied, undefined} serta $r \in R$ berupa three tuple rule (s,o,a) → d. Sebuah *policy* dikatakan sebagai *inconsistency* bila untuk setiap dua buah rule r_i dan $r_j \in R$, dimana $i \neq j$ maka untuk $r_i \rightarrow d_i$ dan $r_j \rightarrow d_j$ dimana dimana $i \neq j$ maka r_i dan r_j akan memberikan hasil *decision* yang kontradiksi.

2. *Incompleteness*

   Yaitu kondisi dimana terdapat *rule* yang belum terakomodasi dalam himpunan *rule* yang sudah didefinisikan sebelumnya. Yaitu terdapat r untuk satu keadaan dimana $r \notin R$.

Tujuan validasi *access control policy* yaitu memastikan tidak terjadinya *inconsistency* dan *incompleteness* di dalam sebuah sistem. Jika masih ada maka sistem keamanan tersebut belum valid atau aman. Kemudian untuk melakukan validasi terhadap *access control policies* menurut Aqib & Shaikh (2015) yaitu memastikan tidak terjadinya *inconsistency* dan *incompleteness*. Dalam hal ini terdapat beberapa metode untuk melakukan validasi yaitu: *Mining technique, model checking technique, formal methods, matrix based approaches, mutation testing* dan *other technique.*

1. *Mining technique*

   Mining technique merupakan teknik yang digunakan untuk mengekstraksi pola data yang berbeda dari sejumlah data yang besar dan mengubahnya menjadi format yang diperlukan untuk membuatnya berguna dalam lingkungan yang berbeda.

2. *Model Checking*

   Model checking merupakan sebuah metode validasi *access control* dengan cara memeriksa elemen-elemen atribut yang ada pada sebuah *access control* sehingga jika terdapat kesalahan maka akan dapat diketahui.

3. *Formal Methods*

   Formal Methods merupakan sebuah metode validasi *access control* dengan melibatkan konsep dan teknik matematika dianggap sebagai metode formal.

4. *Matrix Based Approaches*

   Matrix di dalam matematika biasanya digunakan untuk representasi fungsi liniar dan juga digunakan untuk menemukan solusi dari seperangkat persamaan linear. Sedangkan dalam imu komputer *matrix* biasanya digunakan dalam grafik komputer dimana ini digunakan untuk memproyeksikan gambar dimensi dalam

koordinat dimensi lainnya. Di dalam validasi *access control,* peneliti menggunakan *matrix* ini bekerja sama dengan *tool* lain untuk melakukan validasi *access control.*

5. *Mutation Testing*

   Mutation Testing merupakan pendekatan dan digunakan untuk melakukan pengujian perangkat lunak. Dalam teknik ini kode program yang ada dimodifikasi dengan beberapa cara untuk menghasilkan keluaran yang berbeda dari program asli. Versi modifikasi dari program asli disebut mutan dan outputnya dibandingkan dengan output dari program asli. Jika dua output berbeda, maka mutan dikatakan dibunuh dan output asli diuji terhadap mutan lainnya.

6. *Other Technique*

   Dan beberapa macam teknik lainnya yang telah dilakukan oleh peneliti terdahulu.

Rangkuman metode validasi *access control* dari Aqib & Shaikh (2015) menjadi dasar penulis untuk mencoba menerapkan *model checking* untuk validasi terhadap akses bukti digital. *Model checking methods* menurut Aqib & Shaikh (2015) merupakan sebuah metode yang bersifat memeriksa komponen-komponen di dalam suatu *access control*, dalam hal ini yaitu *policy statement*. *Policy statement* yang ada dalam LPBD tersebut akan diperiksa satu persatu sehingga apabila terdapat ketidaksesuaian pada *policy* tersebut maka akan diketahui kesalahannya sehingga dapat diperbaiki kembali.

Tool untuk membuat ABAC policy yang ada pada saat ini ada beberapa macam, salah satu tool untuk validasi *access control* yaitu *Access Control Policy Testing* atau yang dikenal dengan ACPT (Hwang, Xie, Hu, & Altunay, 2010). ACPT menyediakan 3 fungsi utama yaitu :

1. Membantu menemukan dan menggabungkan *policy* berdasarkan yang sudah dikenal oleh model *policy* yang ada.
2. ACPT menganalisa dan mengonversi sebuah *policy* (dibuat berdasarkan model *policy*) menjadi sebuah format yang dapat di eksekusi seperti XACML.
3. Untuk memastikan bahwa *policy* itu benar, ACPT melakukan statis dan dinamis verifikasi terhadap sebuah *policy*.

ACPT memiliki 3 algoritma untuk membuat dan menguji *ABAC policy statement,* yaitu:

1. *First Applicable*

   First applicable yaitu kondisi disaat *policy statement* yang telah disusun dan diberi hak akses menjadi yang pertama berlaku. Sedangkan *policy statement* yang belum diberi hak akses akan diberikan hak akses yang lain.

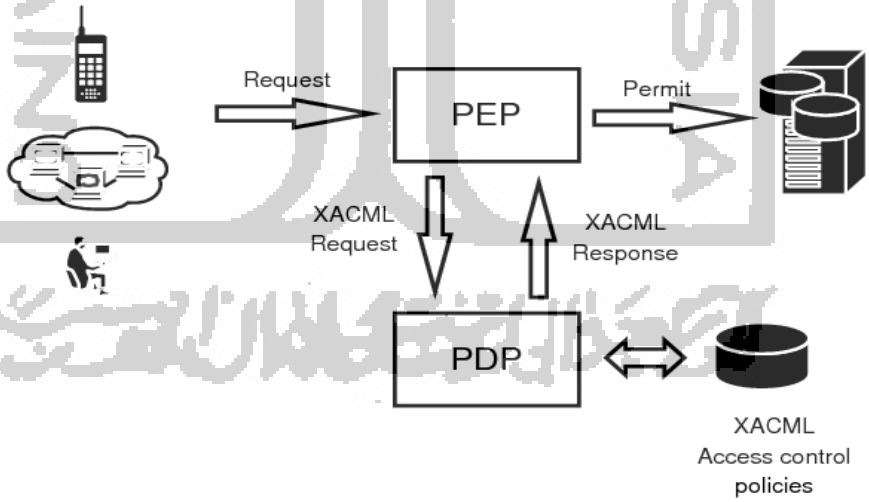
2. *Deny Overrides*

   Deny overrides merupakan algoritma yang menggabungkan keputusan yang sedemikian rupa sehingga jika terdapat keputusan yang *deny*. Maka keputusan itu akan menang.

3. *Permit Overrides*

   Permit Overrides merupakan kebalikan dari *deny overrides* yaitu algoritma yang menggabungkan keputusan sedemikian rupa sehingga jika terdapat keputusan yang *permit*. Maka keputusan itu yang akan menang.

Bahasa yang digunakan untuk membuat *access control* yaitu *Extensible Access Control Markup Language* (XACML). XACML merupakan berbasiskan pada standar XML yang dirancang untuk membuat *access control.* XACML dijadikan standar oleh Organisasi untuk Kemajuan Standar Informasi Terstruktur (OASIS) untuk mendefinisikan arsitektur, *policy* dan pesan dalam *access control*. Framework XACML menurut Mankai & Logripppo (2005) dapat dilihat pada gambar di bawah ini:



Gambar 2.2 Framework XACML (Mankai & Logripppo, 2005)

Dari gambar di atas menunjukan model XACML yang berisi dua entitas utama yaitu *PEP (Policy Enforcement Point)* dan *PDP (Policy Decision Point).* PEP yaitu entitas yang melindungi sumber daya. PEP menerima permintaan akses dan meneruskannya ke PDP.

PDP membuat keputusan sesuai dengan informasi yang terkandung dalam permintaan dalam konteks XACML. Setiap permintaan mendefinisikan *subject*, sumber daya dan rindakan yang ditandai oleh satu set atribut untuk mendefinisikan sifat mereka.

Access control policy disimpan dalam bentuk file xml dan disusun menjadi *rule*, *policy* dan *policy set*. Beberapa *rule* dikelompokkan kedalam *policy* dan *policy* dikelompokkan kedalam *policy set*. *Rule*, *policy* dan *policy set* menentukan target yang menunjukan domain penerapannya. Sedangkan target menentukan satu set atribut yang telah diberi nilai sesuai dengan *request*.