IBM Research / Linux Technology Center

Implementing GSER and Component Matching

Sang Seok Lim
IBM Research

Jong Hyuk Choi
IBM Research

Kurt Zeilenga
OpenLDAP

© 2004 IBM Corporation

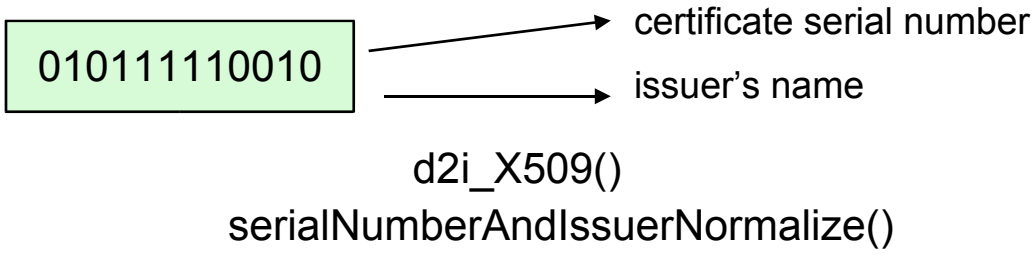
Contents

When LDAP meets PKI

- **LDAP when used to support Public Key Infrastructures**
  - Search for the correct Certificate or Certificate Revocation List (CRL) based on their contents
    - "Find *userCertificate* attribute containing the email address slapd@openldap.org"
  - Store and transfer a certificate as binary blobs
    - No knowledge about structure and contents
- **Three approaches to handle the assertion**
  - Syntax specific parsing
  - Attribute extraction
  - Component matching : our approach

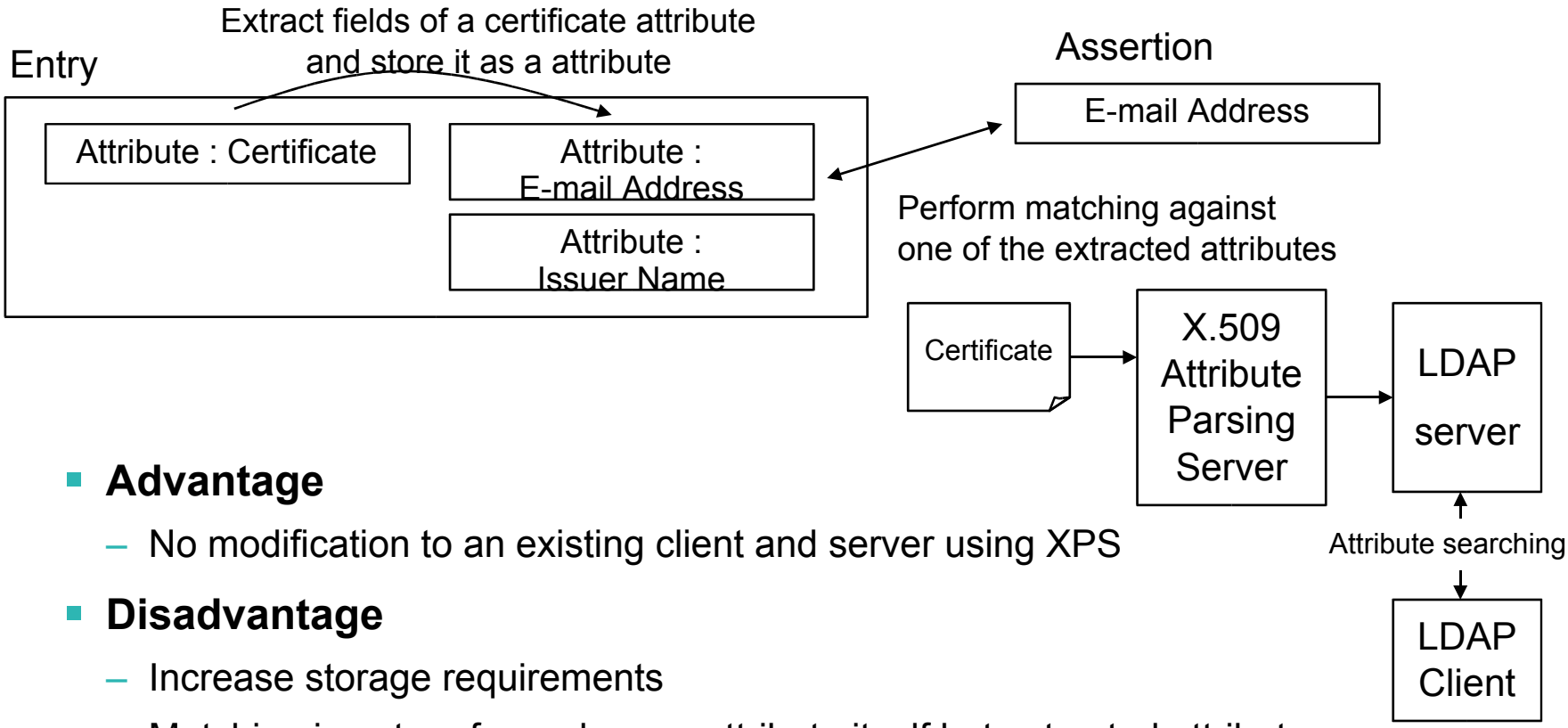
Syntax Specific Parsing

- **Parsing the blobs to recognize the fields of X.509 ASN.1**



- **Disadvantages**
  - The parsing routine is dedicated to a specific syntax (X.509 Certificate)
    - rewrite syntax specific routines whenever a new ASN.1 syntax and any change in existing ASN.1 come up

Attribute Extraction



- **Advantage**
  - No modification to an existing client and server using XPS
- **Disadvantage**
  - Increase storage requirements
  - Matching is not performed on an attribute itself but extracted attributes
    - Security issue
  - Complicated mechanism to support multiple certificates of an user

Component Matching

- **The Automated and fundamental approach of syntax specific parsing**
- **Define a generic way of matching user selected components in an attribute value of ASN.1 attribute syntax**
  - Attribute Syntax of X.500 and LDAP
    - Described by Abstract Syntax Notation (ASN.1) type definitions
  - Component assertion, component filter , *componentFilterMatch* matching rule
  - GSER : represent a component assertion value as regular and uniform UTF8 string encoding

Component Matching

- **Advantages**
  - X.509 attribute is stored only once
  - Matching is performed directly on their contents
  - Easy to deploy a new syntax and matching rule with the help of an ASN.1 compiler
- **Disadvantages**
  - Modifications to an LDAP server
    - the new matching rules (*componentFilterMatch*) and GSER

GSER : ASN.1 encodings (RFC 3641)

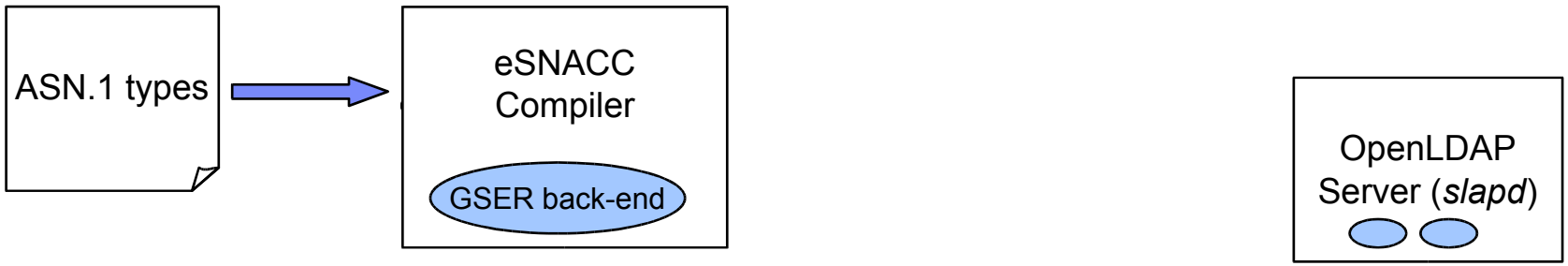
- **A human readable UTF-8 character string encoding of ASN.1 values of any give arbitrary ASN.1 type**
- **Define** *UTF8 string encodings* **at the lowest level of ASN.1 built-in types**
  - INTEGER, BOOLEAN, OCTET STRING etc
- **Build up complex ASN.1 types** automatically **from the lowest level**
  - SEQUENCE, SET, CHOICE

```
TBSCertificate  :: = SEQUENCE {
    version          [0] EXPLICIT Version DEFAULT v1,
    serialNumber     CertificateSerialNumber,
    signature        AlgorithmIdentifier,
    issuer           Name,
    validity         Validity,
    subject          Name,
    subjectPublickKeyInfo subjectPublicKeyInfo,
      …
    extensions   [3] EXPLICIT Extensions OPTIONAL
}
```

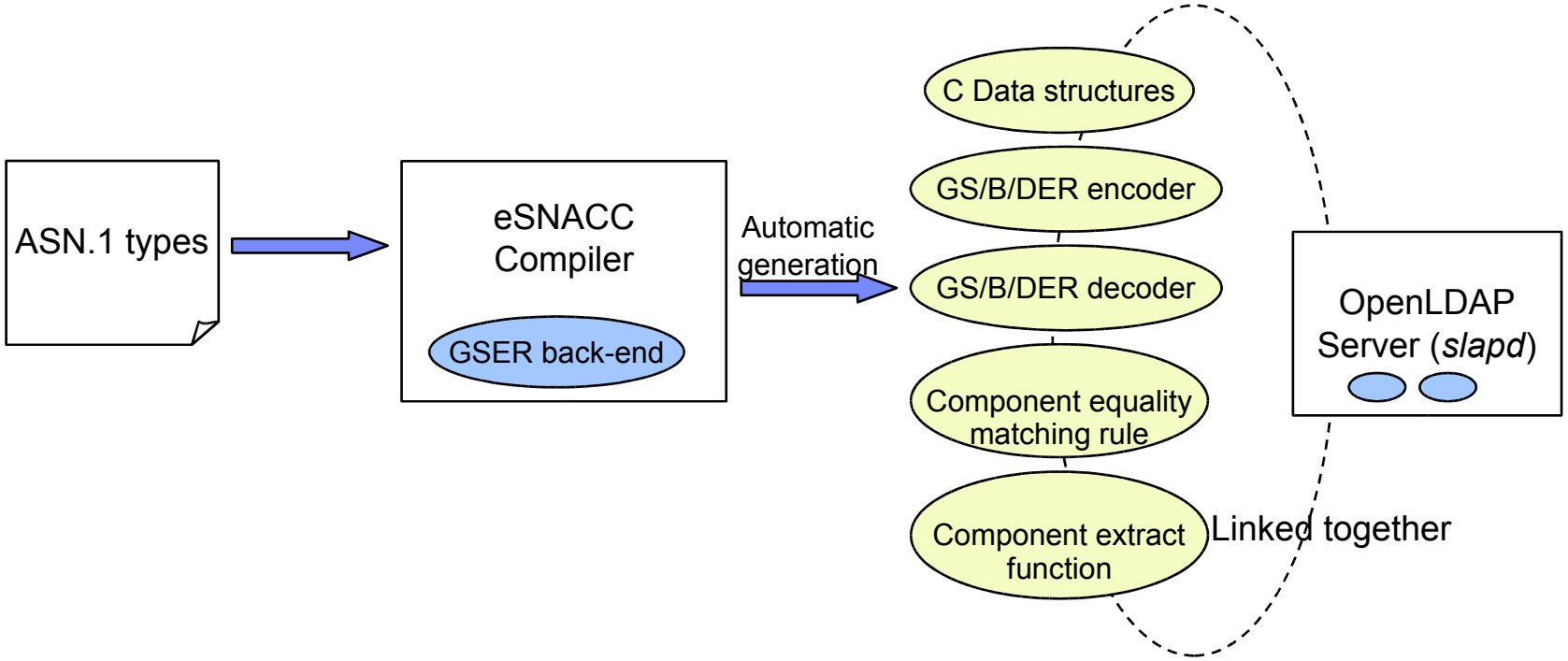
{ version v1, serialNumber 12345 ,signature {… },
"issuer cn=foobar,o=ibm" … }

GSER encodings

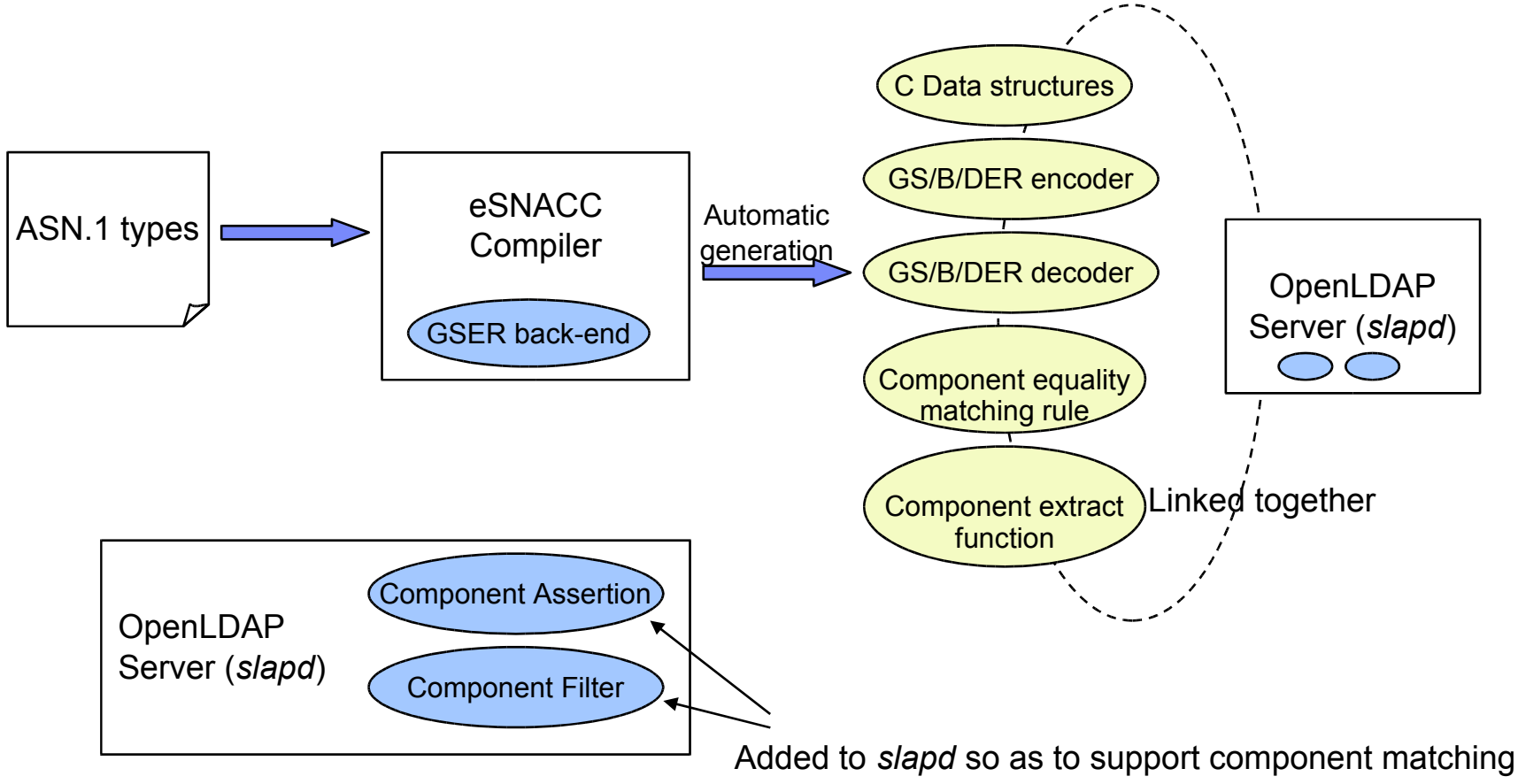
Overview of Component Matching enabled OpenLDAP



Overview of Component Matching enabled OpenLDAP



Overview of Component Matching enabled OpenLDAP

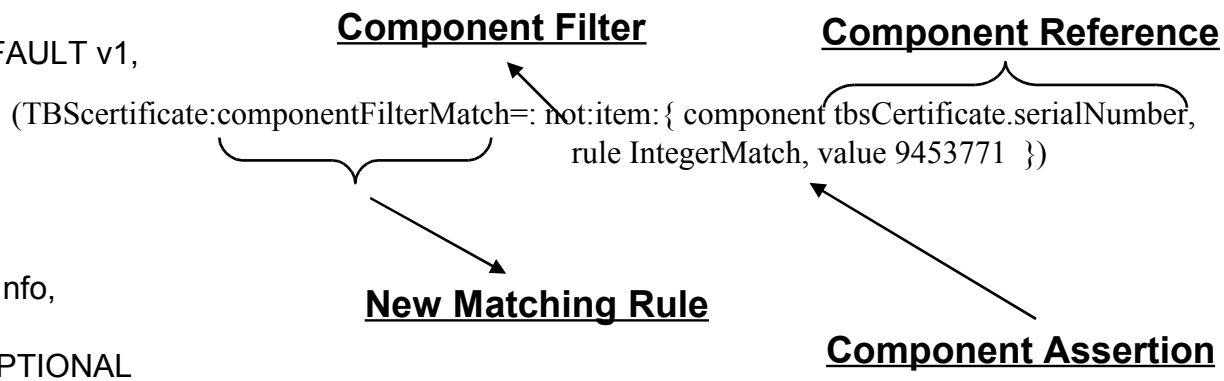


We will fully automate overall steps required for ASN.1 syntax registration and activation

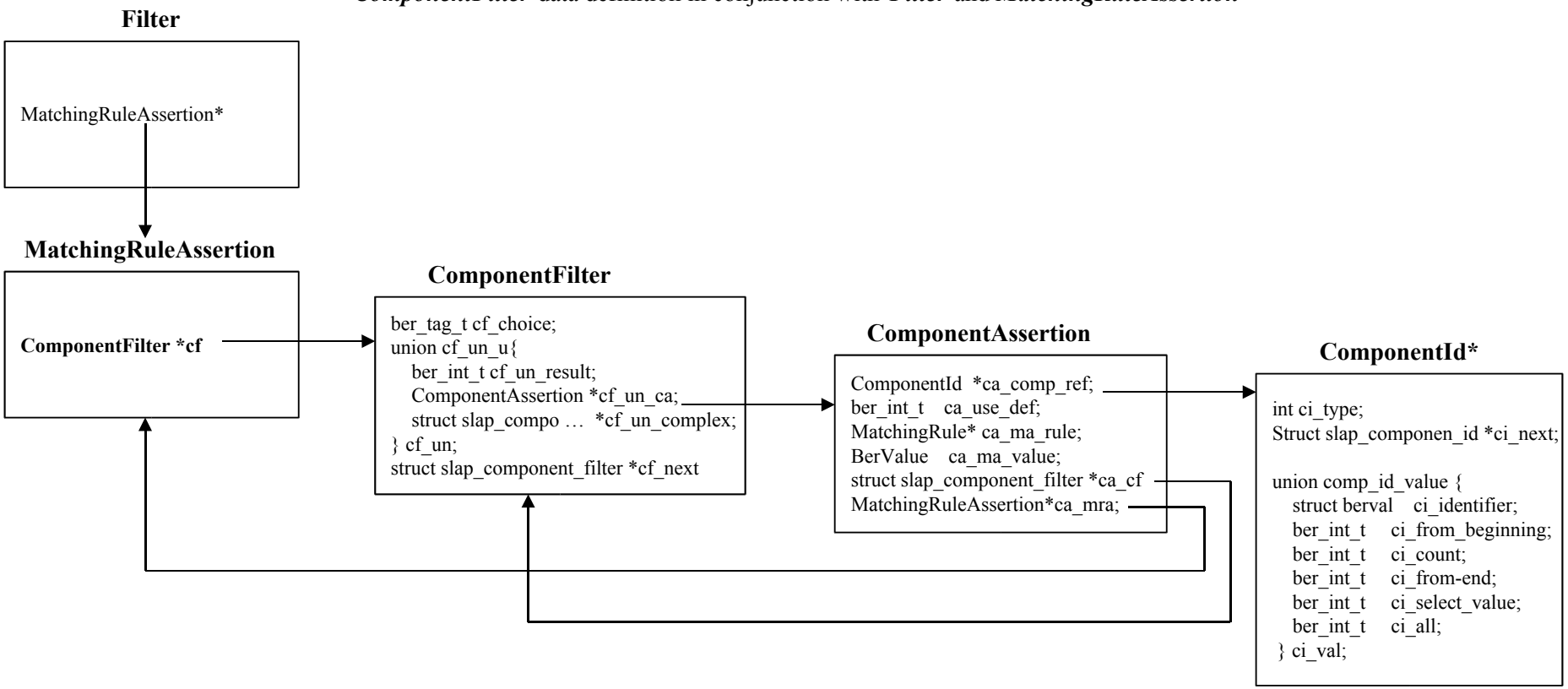
Component Assertion and Component Filter

- **Component Assertion**
  - An assertion about the presence, or values of, components within an ASN.1 value
  - Component Reference
    - identifying the component part of a ASN.1 value.
- **Component Filter**
  - An “and”, “or”, “not” expression of ComponentAssertion, evaluates to either TRUE, FALSE or Undefined

```
TBSCertificate  :: = SEQUENCE {
    version          [0] EXPLICIT Version DEFAULT v1,
    serialNumber  CertificateSerialNumber,
    signature        AlgorithmIdentifier,
    issuer           Name,
    validity         Validity,
    subject          Name,
    subjectPublickKeyInfo subjectPublicKeyInfo,
      …
    extensions   [3] EXPLICIT Entensions OPTIONAL
}
```



ComponentFilter data definition in conjunction with ***Filter*** and ***MatchingRuleAssertion***

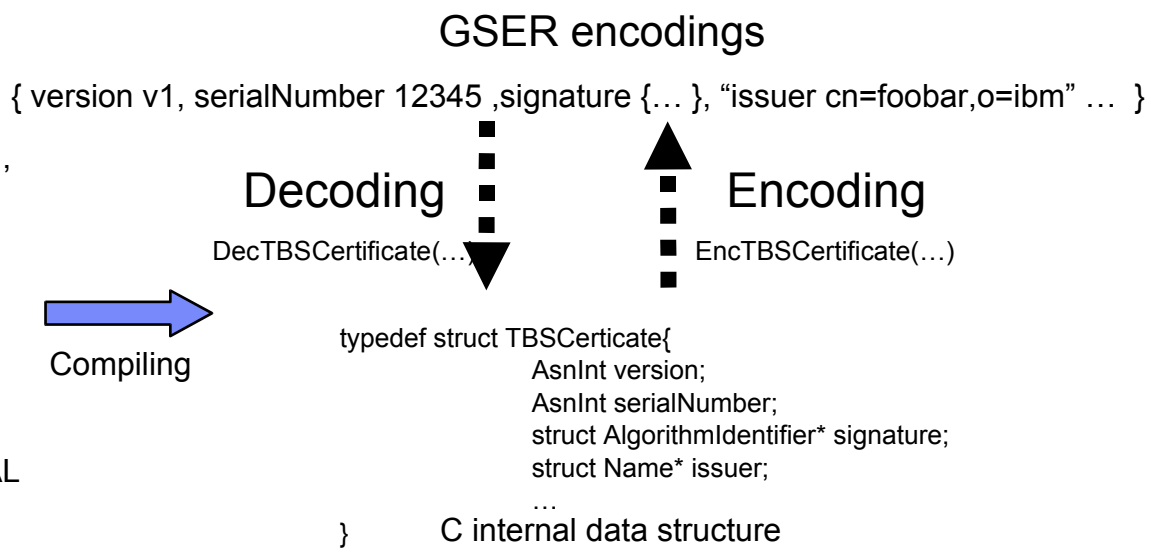


An eSNACC compiler and an GSER back-end

- **What does the compiler do?**
  - Compile ASN.1(Abstract Syntax Notation One) modules into
    - ASN.1 equivalent C data structures
    - Routines to convert values between the internal (C or C++) representation and the corresponding BER/DER formats
- **GSER back-end**
  - Generate GSER encoding/decoding functions of given ASN.1 definition

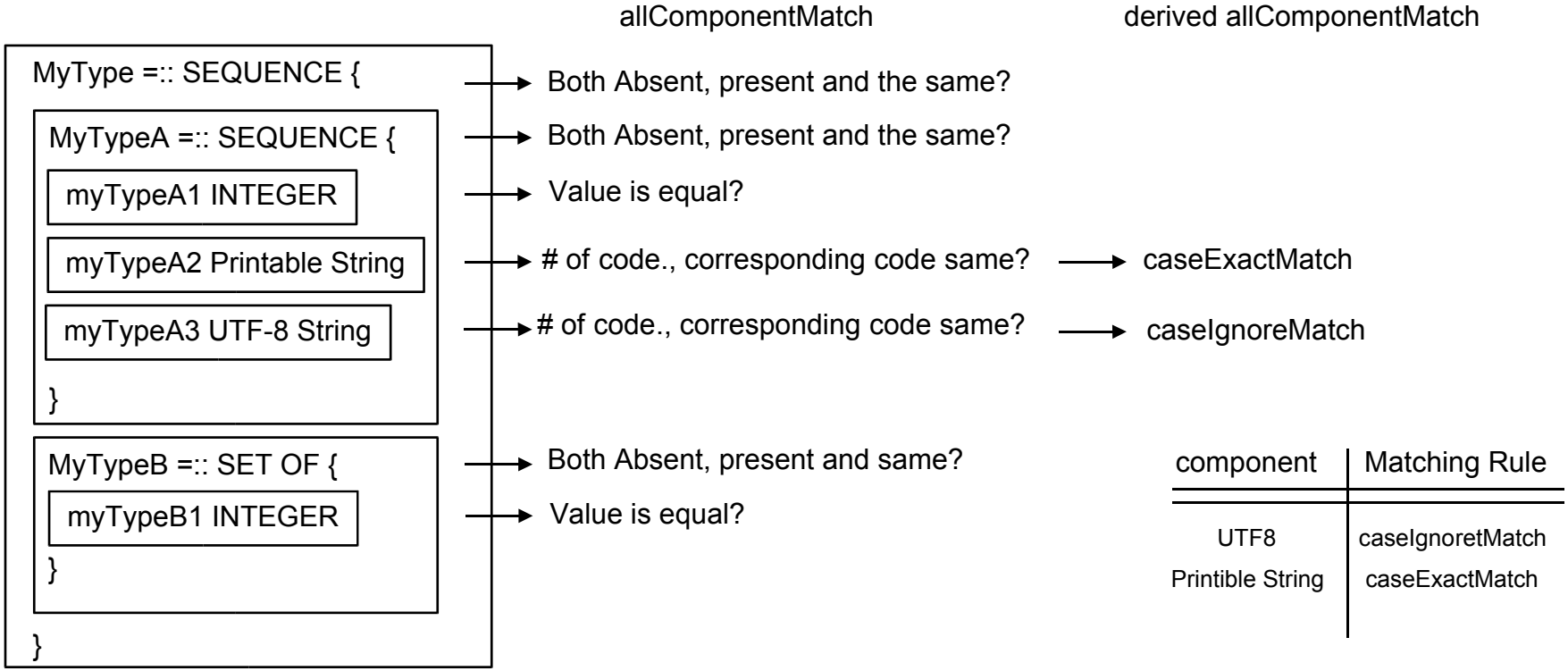
```
TBSCertificate  :: = SEQUENCE {
    version          [0] EXPLICIT Version DEFAULT v1,
    serialNumber  CertificateSerialNumber,
    signature        AlgorithmIdentifier,
     issuer           Name,
     validity         Validity,
     subject          Name,
     subjectPublickKeyInfo subjectPublicKeyInfo,
      …
     extensions  [3] EXPLICIT Extensions OPTIONAL
}
```

Certificate ASN.1 type



Component Equality Matching Rule (1)

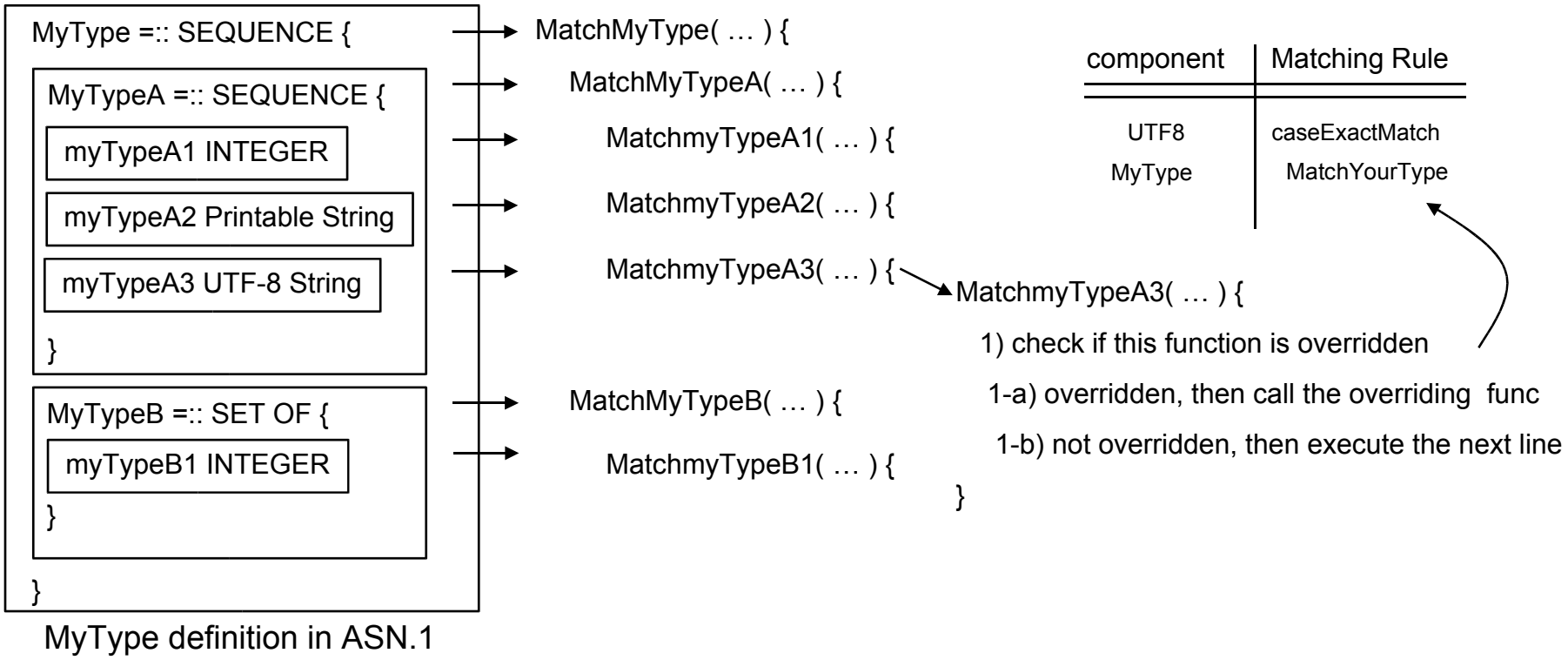
- **Equality matching rule for a complex ASN.1 type**
  - allComponentMatch : basic equality matching rule for an ASN.1 type
  - derived allComponentMatch : refined equality matching rule



MyType definition in ASN.1

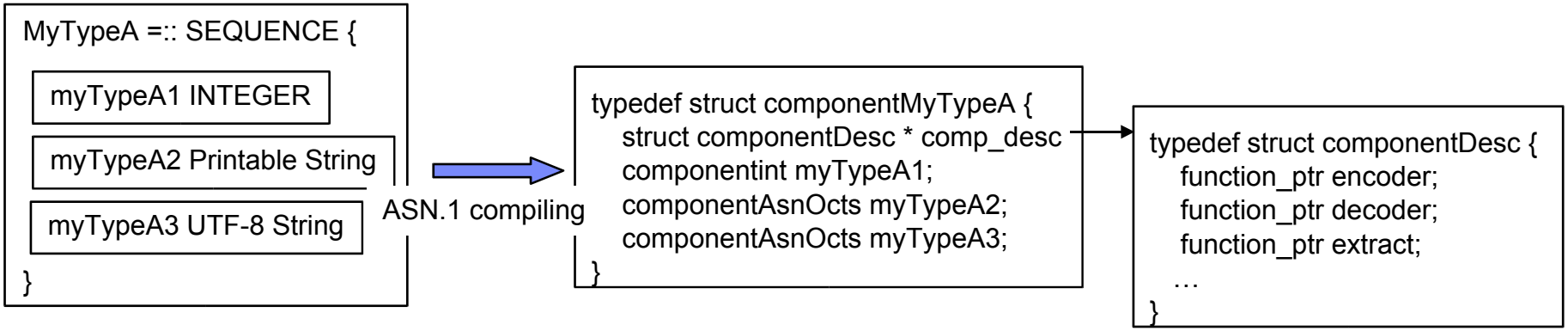
Component Equality Matching Rule (2)

- **Automatic generation of *allComponentMatch* using eSNACC**



Component Representation in *slapd*

- **Component**
  - The selected part of attribute value
    - referenced or identified by Component Reference
  - Two parts
    - Data value : Value in a C internal data structure
    - Descriptor for its value processing : value-specific encoder/decoder/matching rule/extract
- **Component extraction**
  - Extract referenced components from a component tree



```
typedef struct componentMyType {
    struct componentDesc *
comp_desc
    component* myTypeA;
    component* myTypeB;
}
```

```
typedef struct componentDesc
{
    function_ptr encoder;
    function_ptr decoder;
    function_ptr extract;
    function_ptr matching;
}
```

MyType syntax in ASN.1

```
MyType =:: SEQUENCE {
    MyTypeA =:: SEQUENCE {
        myTypeA1 INTEGER,
        myTypeA2 Printable String,
        myTypeA3 UTF8 String,
    },
    MyTypeB =:: SET OF {
        myTypeB1 INTEGER
    }
}
```

```
typedef struct componentMyTypeA {
    struct componentDesc * comp_desc
    componentInt myTypeA1;
    componentPrintStr myTypeA2;
    componentUTF8  myTypeA3;
}
```

```
typedef struct componentDesc {
    function_ptr encoder;
    function_ptr decoder;
    function_ptr extract;
    function_ptr matching;
```

```
typedef struct componentMyTypeB {
    struct componentDesc * comp_desc
    componentInt myTypeB1;
    }
```

Composite ASN.1 type

```
typedef struct componentDesc {
    function_ptr encoder;
    function_ptr decoder;
    function_ptr extract;
    function_ptr matching;
}
```

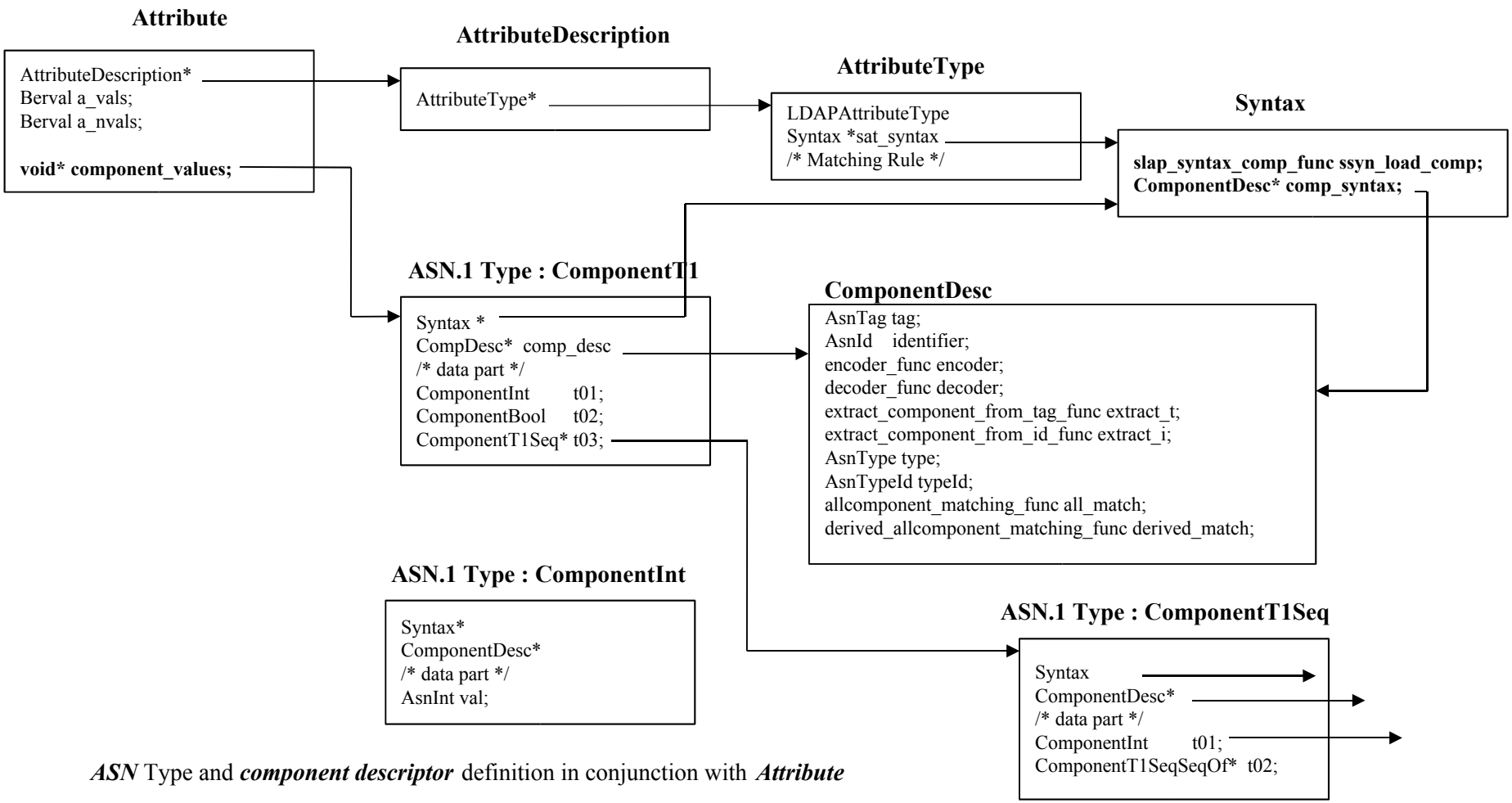
```
typedef struct slap_attr  {
    AttributeDescription *a_desc;
    BerVarray a_vals;
    BerVarray a_nvals;
    ….
    } Attribute
```

Attribute definition in slapd

```
typedef struct componentInt {
    struct componentDesc * comp_desc
    AsnInt  value;
    }
```

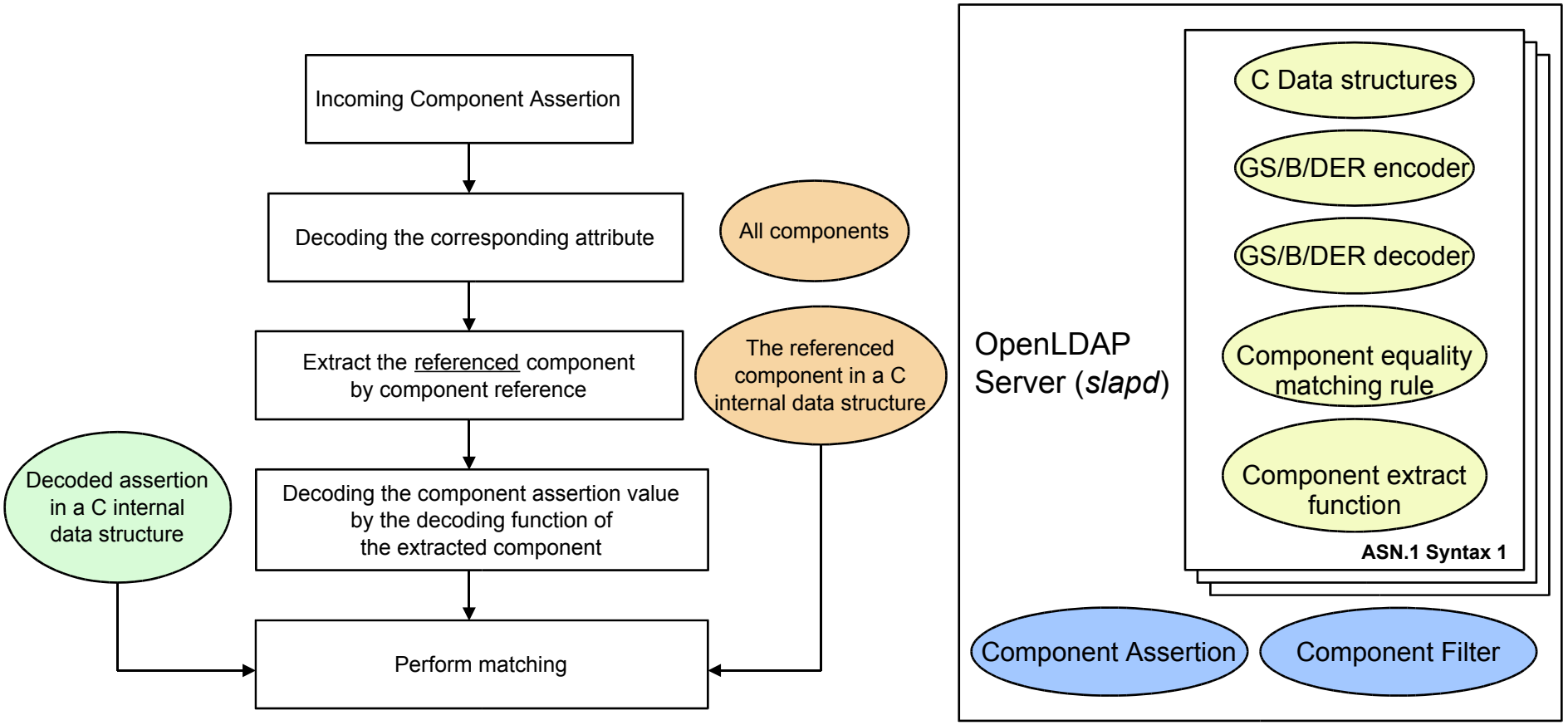
Basic ASN.1 type

```
typedef struct componentDesc {
    function_ptr encoder;
    function_ptr decoder;
    function_ptr extract;
    function_ptr matching;
}
```



ASN Type and ***component descriptor*** definition in conjunction with ***Attribute***

Overall Matching Flow of Component Matching Enabled OpenLDAP



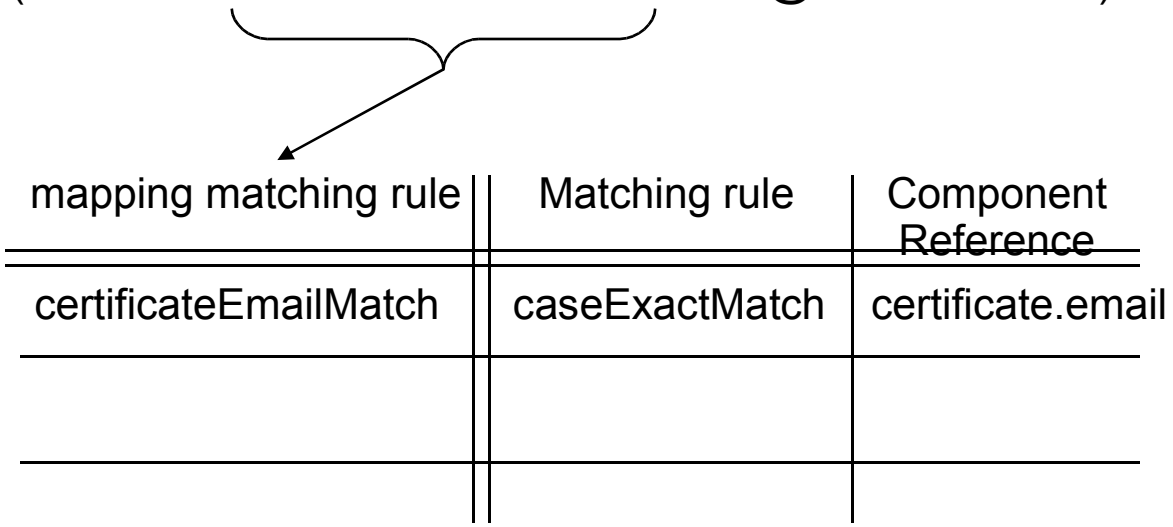
Applicability

- **GSER and component matching**
  - Generic way of defining a syntax and matching rule
- **Component matching enabled OpenLDAP server**
  - Web Service
    - WS-security based on SOAP Message Security
    - Security token reference : <wsse:Reference URI="ldap://xx.com/CN=foo.."/>
      - Token: name, identity, key , group etc
  - Grid and On-Demand computing
    - Grid Security Infrastructure based on public key technologies
    - Grid object specification : represent compute resources
      - Easy to deploy new schemas by the help of eSNACC compiler

Mapping attribute matching to component matching

- **When a client does not support GSER and extensible matching and a server supports component matching**
- **Attribute matching to component matching**
  - Using attribute matching OID as a key, find matching rule to be performed instead

(Certificate::certificateEmailMatch=sslim@core.kaist.ac.kr)



mapping matching rule	Matching rule	Component Reference
certificateEmailMatch	caseExactMatch	certificate.email

Summary

- **Need to match a selected part of an attribute value**
  - ASN.1 aware OpenLDAP server
  - GSER and Component matching : automated and fundamental approach
- **Component matching enabled OpenLDAP server**
  - Component assertion, filter and reference of component matching
  - eSNACC compiler to automate syntax writing and deployment
    - GSER encoding/decoding/equality matching rule/component extraction routines
  - Component representation in slapd
- **Question?**

References

- **RFC 3687 : LDAP and X.500 Component Matching Rules**
- **Deficiencies in LDAP when used to support Public Key Infrastructures**
- **ASN.1 : Abstract Syntax Notation One**
- **RFC 3641 : Generic String Encoding Rules (GSER) for ASN.1 Types**