Information Network 1
Application layer protocols

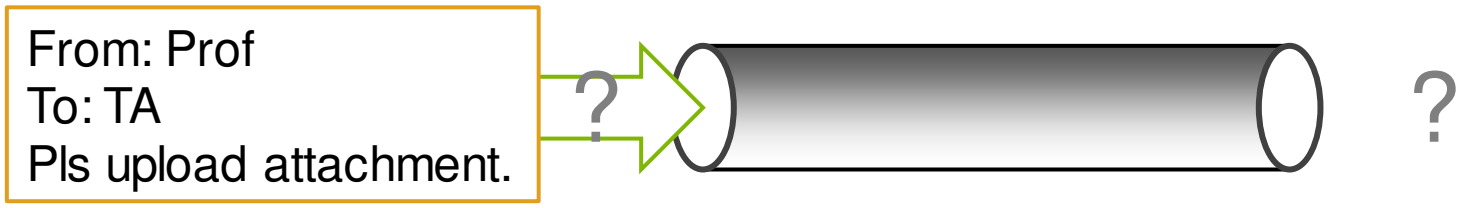
2013/5/31

Youki Kadobayashi

NAIST 奈良先端科学技術大学院大学

Application layer protocol challenges

- How do I interact with the other end?
- How do I find the other end?
- How do I send information?
  - Can I directly use TCP?



How do I start interacting with the other end?

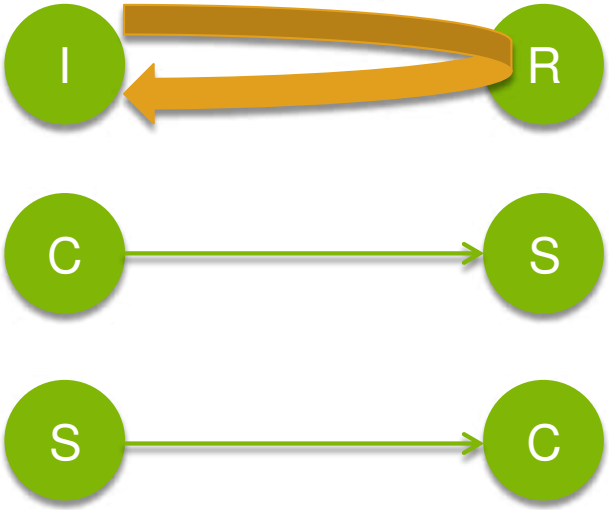
- Who will initiate the interaction?
- Who will be responder?

“request” “response”

“query” “response”

Initiator-Responder vs Client-Server

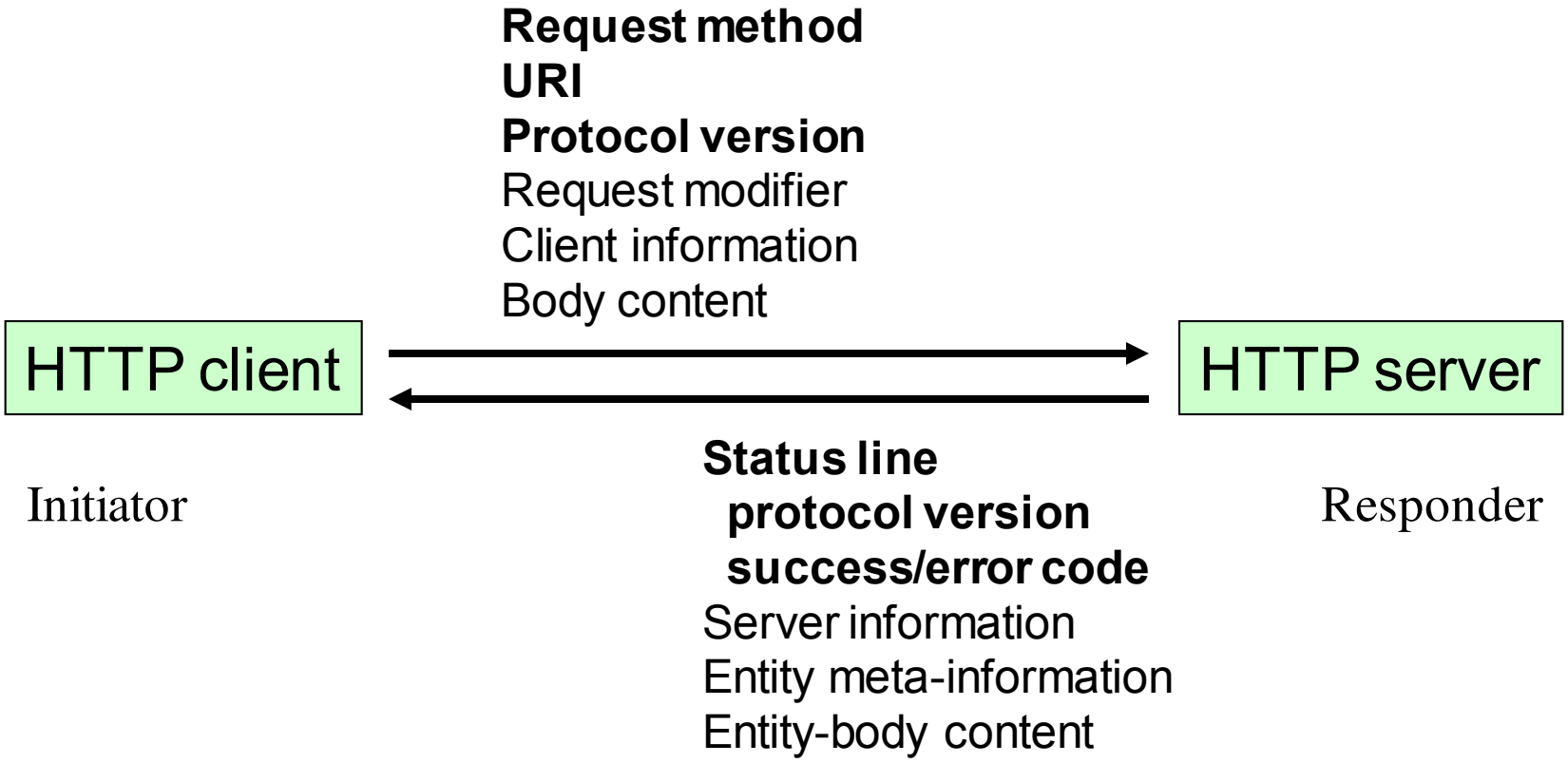
- Initiator can be either client or server
- Responder can be either server or client



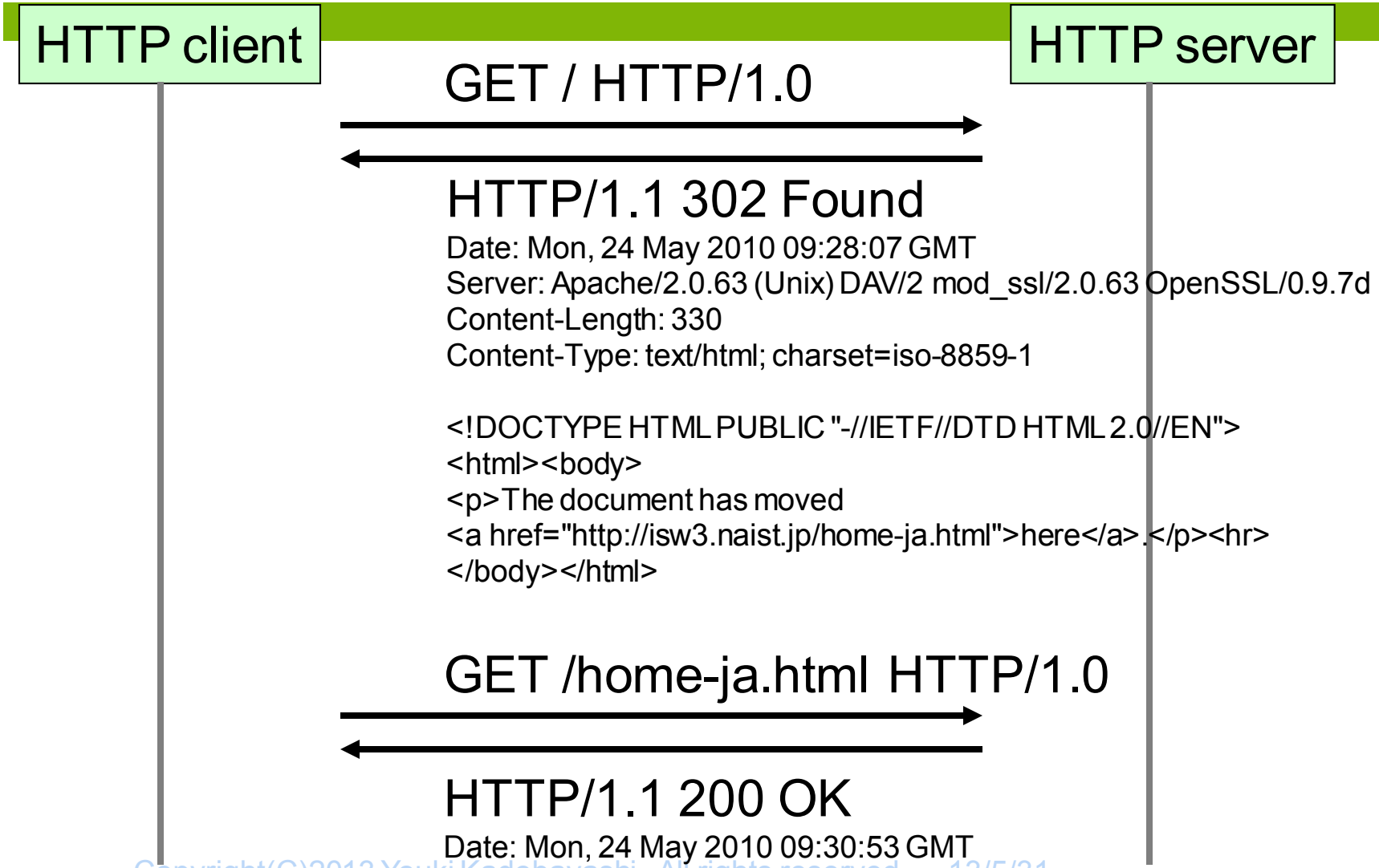
Consider an application-layer protocol for Chat app.
If only Clients can be initiator, how do we deliver message to your friends?

The web protocol: HTTP

- RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1

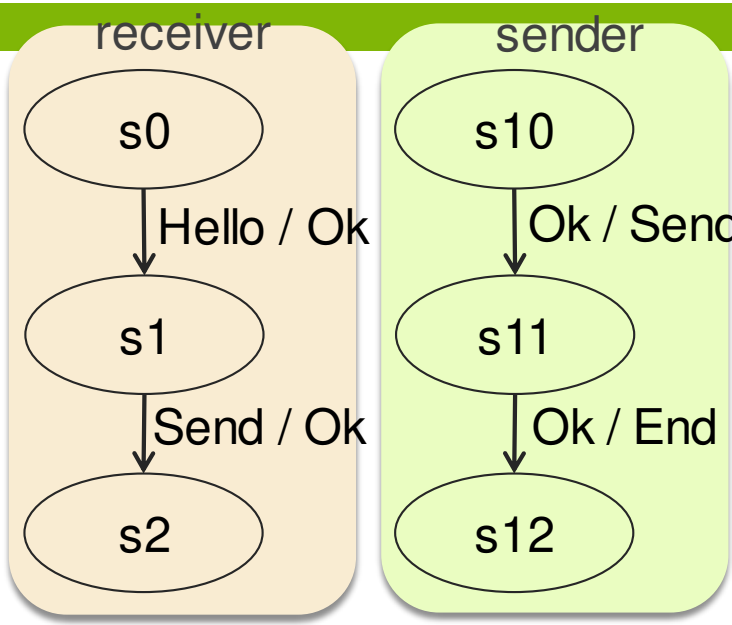


HTTP in action



How do I interact with the other end?

- What kind of information:
  - Will be sent?
  - Should be received?
- State transitions?



- Essentially you are designing a protocol FSM: finite state machine.
- Protocol FSM may deadlock, without careful design.

Ensuring desirable protocol properties

- Liveness property
  - Something good eventually happens
  - e.g., completion of data transmission
- Safety property
  - Something bad never happens
  - e.g., starvation
- → model checker

Ensuring desirable protocol properties with model checker


```
chan q[2] = [MaxSeq] of { byte, byte };   /* message passing channel */

active [2] proctype p5()     /* starts two copies of proctype p5 */
{ byte NextFrame, AckExp, FrameExp, r, s, nbuf, i;
  chan in, out;
  in = q[_pid];
  out = q[1-_pid];
  xr in; xs out;          /* partial order reduction claims */

  do
  :: nbuf < MaxSeq ->    /* outgoing messages */
       nbuf++;
       out!NextFrame,    (FrameExp + MaxSeq) percent (MaxSeq + 1);
       inc(NextFrame)

  :: q[_pid]?r,s ->        /* incoming messages */
       if
       :: r == FrameExp ->
            printf("MSC: accept percentd\n," r);
            inc(FrameExp)
       :: else /* ignore message */
       fi;
       do
       :: ((AckExp <= s) && (s < NextFrame))
       || ((AckExp <= s) && (NextFrame < AckExp))
       || ((s < NextFrame) && (NextFrame < AckExp)) ->
             nbuf--;
             inc(AckExp)
       :: else -> break
       od
```

Appendix C: Verification model of a sliding window protocol

For further study:

- **LTL**
- **Model checking**

Source: The model checker SPIN, IEEE Trans. on Software Eng., May 1997

Q&A

How do I find the other end?

Three well-known patterns of finding the peer:

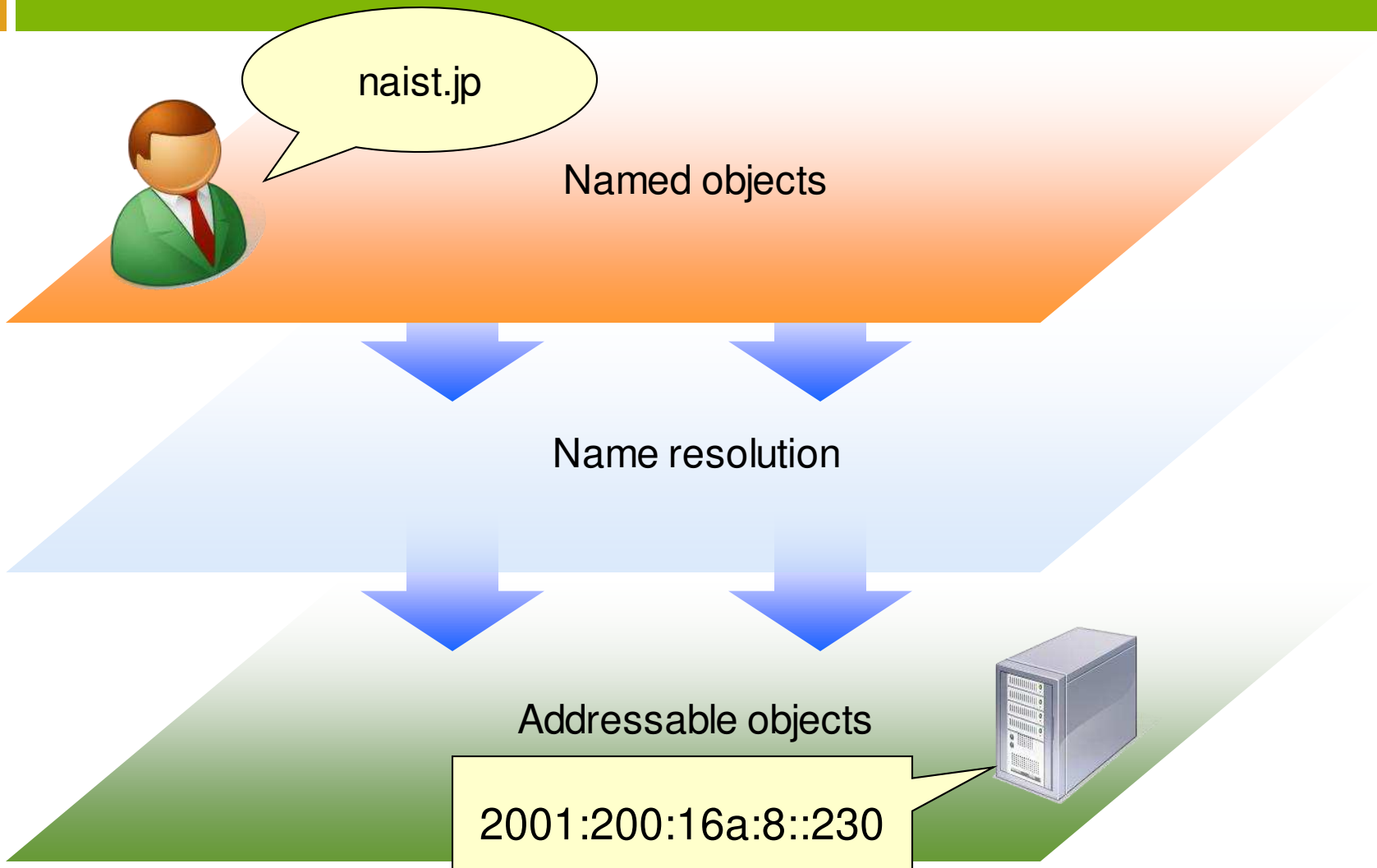
- 1. the other end is known *a priori*.
- 2. central authority knows the other end.
- 3. the other end has to be somebody who processes this information.

1. *A priori* knowledge of endpoint

- A name or IP address is provided in advance..
  - www.somecompany.com
  - 123.4.56.78
- e.g., via Uniform Resource Locator (RFC1738)
  - Type of application protocol
  - TCP port number
  - Host name
  - etc.

```
      http://example.com:8080/over/there?name=ferret#nose
       ¥_/   ¥_____________/¥_________/ ¥_________/ ¥__/
        |           |            |           |        |
     scheme     authority      path        query   fragment
```

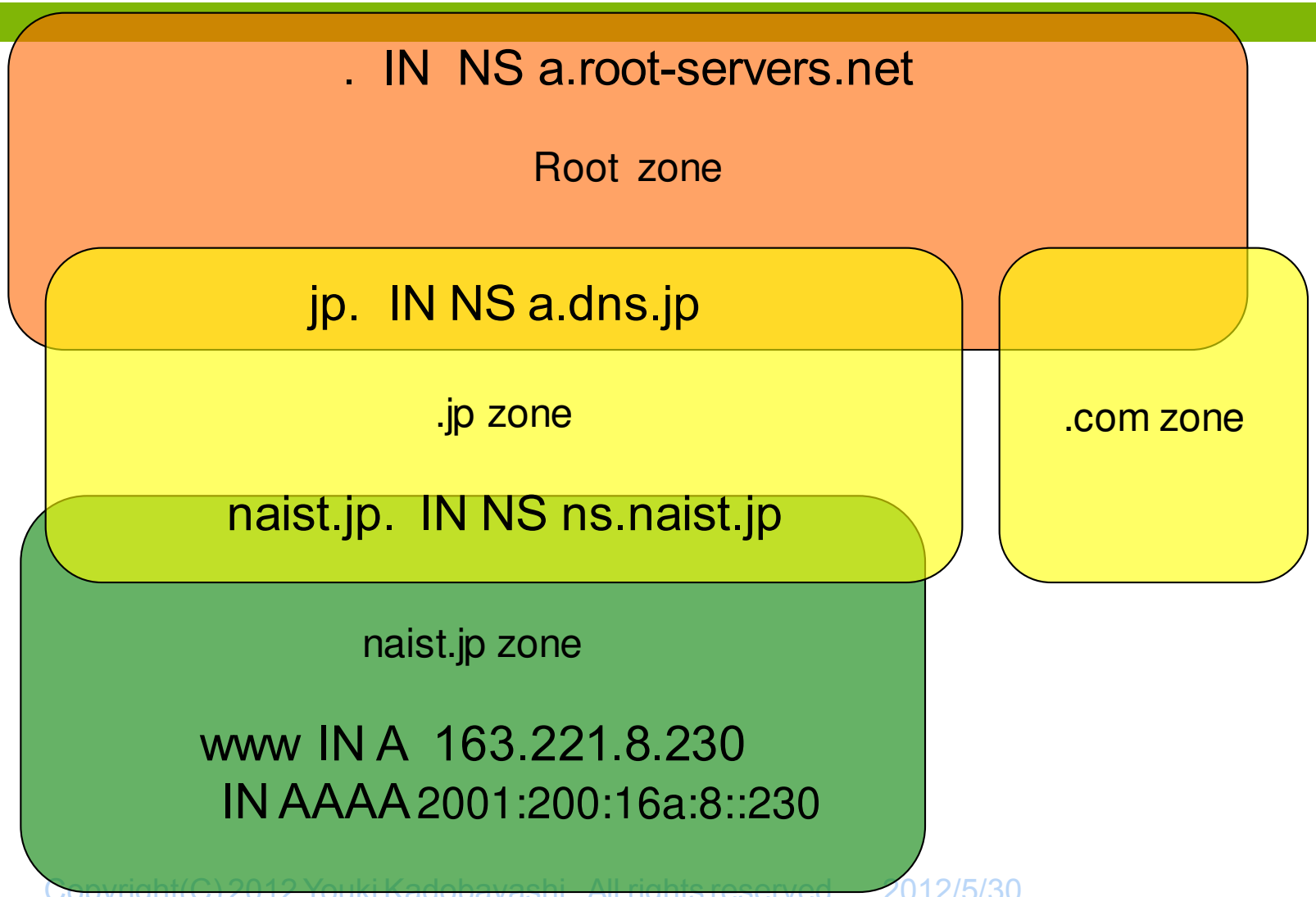

Addressing vs naming: its resolution



DNS: domain name system

- Single global tree for hostname to IP resolution
- Hierarchical delegation
  - Root
    - .
  - gTLD: generic top-level domain
    - .com .net .org .biz .info …
  - ccTLD: country code top-level domain
    - .jp .th .ch .fr …
  - Secondary level domains
    - co.jp, ac.jp, go.jp …

DNS: distributed DB of records



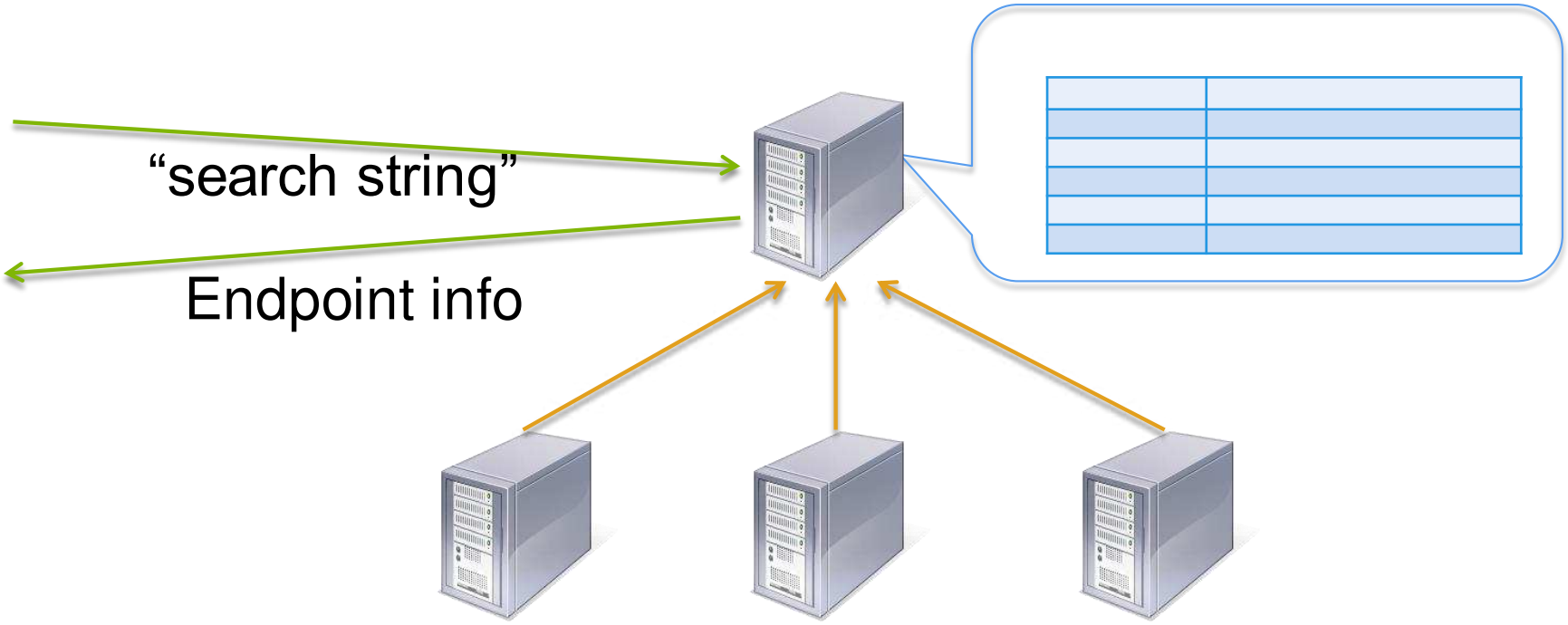
Hands on: observe DNS in action

In the provided virtual machine:

- Run wireshark
- Visit some websites
- Observe DNS protocol interactions

- Some DNS jargons:
  - NS: name server
  - A: address
  - AAAA: IPv6 address

2. A central authority for endpoint lookup



- Many algorithms for scaling beyond 1000 servers
- For further study: NSDI, SIGCOMM papers

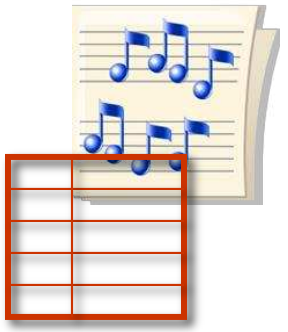
3. Discovery of endpoints from information types

- Discovery of services associated with the resource

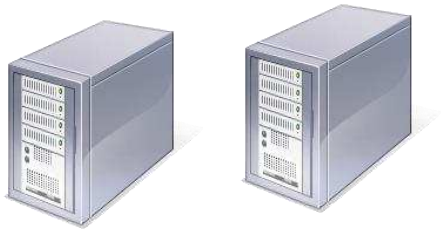
Resource identifier

Resolver

Service endpoint identifier,
Capabilities



?



Modern example: OpenID

Q&A

How do I send information?

- Numeric 1 can be represented in many ways:
  - 00000001
  - 65 (ASCII code)
  - U+0041 (Unicode)
  - Type=1 Length=2 Value=1

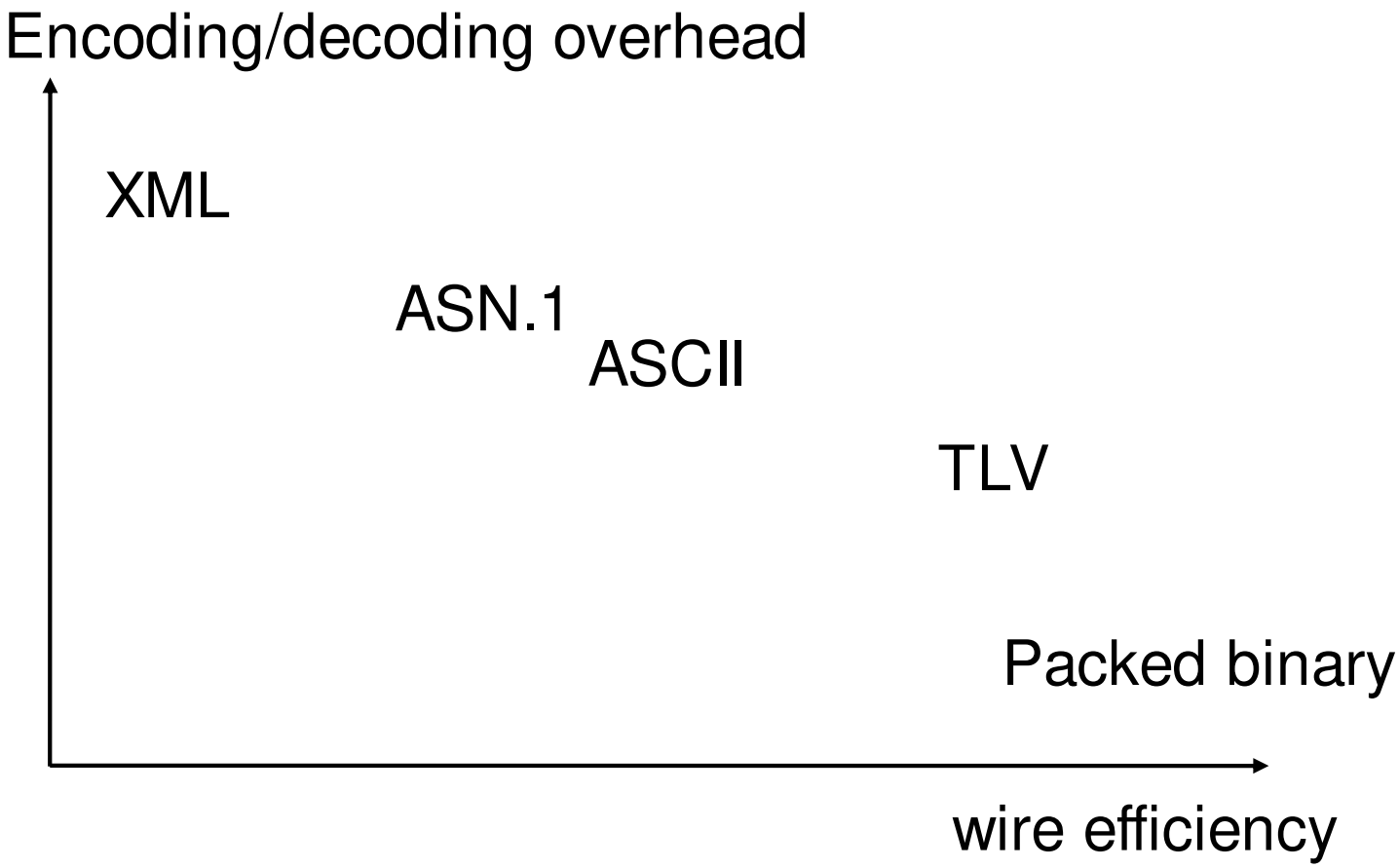
- These are encoding practices defined in the Presentation layer

Established encoding practices in the presentation layer

- Binary
  - 00000001
- ASCII
  - 65
- TLV
  - Type=1 Length=2 Value=1

- ASN.1
  - → network management
- XML

Trade-offs in data representation



Encoding application protocols

Design choices

- Packed binary
- ASCII
- TLV

- JSON
- XML
- ASN.1

… impacting:

- Efficiency
- Extensibility
- Readability
- Ease of definition
- Ease of parsing
- Ease of mixing
- Ease of validation

Packed binary

- Define bit boundaries
- Define semantics associated with each bit

- IP header (RFC791), TCP header (RFC793)
- Same technique can be employed in application layer, e.g., for sensor networks

16bit source port			16bit destination port
32bit sequence number			
32bit acknowledgment number			
4bit hlen	reserved	flags	16bit window size
16bit TCP checksum			16bit urgent pointer

20 octets

ASCII

- Consider byte-stream of ASCII characters as a protocol
  - Point: make it human readable and writable
- ASCII protocols in the Internet
  - Mixture of machine-parseable status code and human-readable strings
- FTP (RFC959), SMTP (RFC2821) etc.
  - 3-digit status code, followed by greetings
  - 4-character command, followed by arguments

FTP: File Transfer Protocol

01.204.01754: 220 ftp.isi.edu NcFTPd Server (free educational license) ready.
76.020.00021: USER anonymous
01.204.01754: 331 Guest login ok, send your complete e-mail address as password.
76.020.00021: PASS -wget@
01.204.01754: 230 Logged in anonymously.

76.020.00021: SYST
01.204.01754: 215 UNIX Type: L8
76.020.00021: PWD
01.204.01754: 257 "/" is cwd.
76.020.00021: TYPE I
01.204.01754: 200 Type okay.
76.020.00021: CWD /in-notes
01.204.01754: 250 "/in-notes" is new cwd.

TLV

- Type, Length, Value
  - Pros:
    Compact, yet extensible
  - Cons:
    All data types have to be defined in protocol spec

- RADIUS (RFC2138), OSPF (RFC2328) etc.
  - Bit representation of TLV

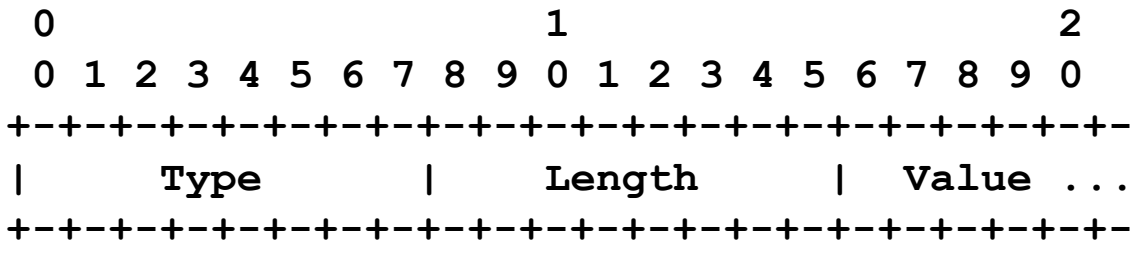
RADIUS Packet Format

- (RFC2138) 3. Packet Format

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     Code       |  Identifier   |            Length             |
-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                                                               |
                          Authenticator                        |
                                                               |
                                                               |
-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  Attributes ...
-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

ode:
        Access-Request
        Access-Accept
        Access-Reject
```

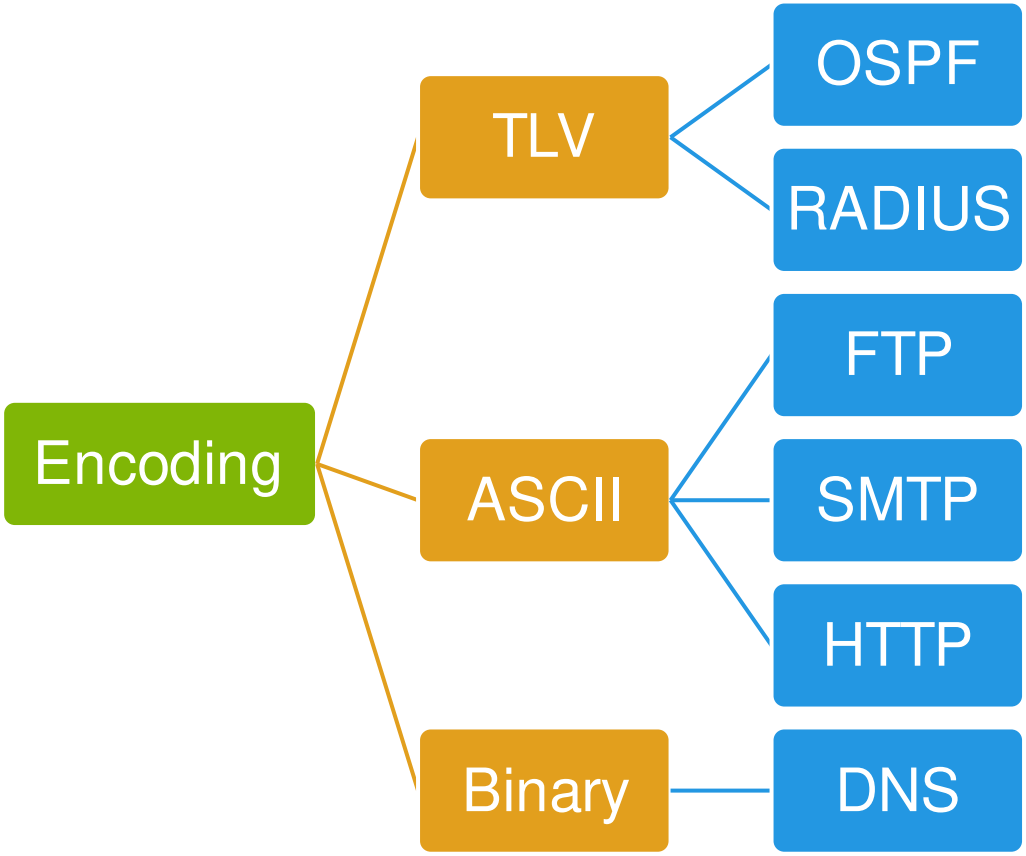

RADIUS Attribute TLV



Type

1	User-Name
2	User-Password
3	CHAP-Password
4	NAS-IP-Address

A taxonomy of application layer protocols

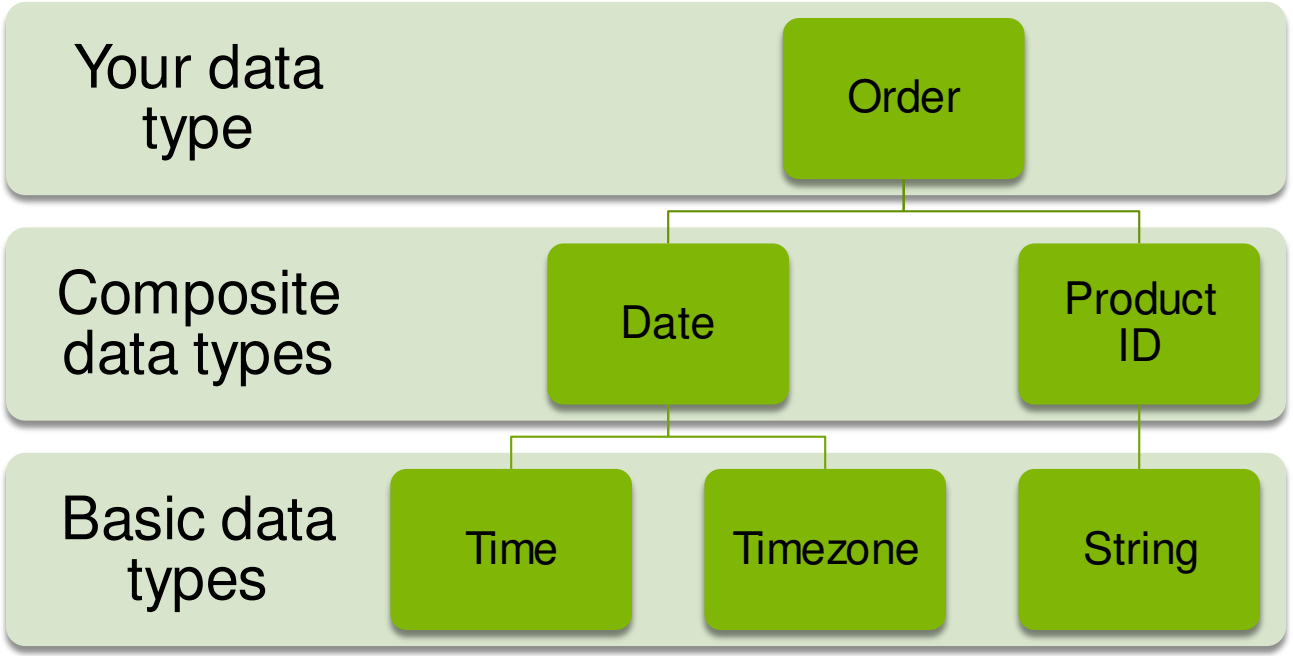


Q&A

XML: Extensible data types

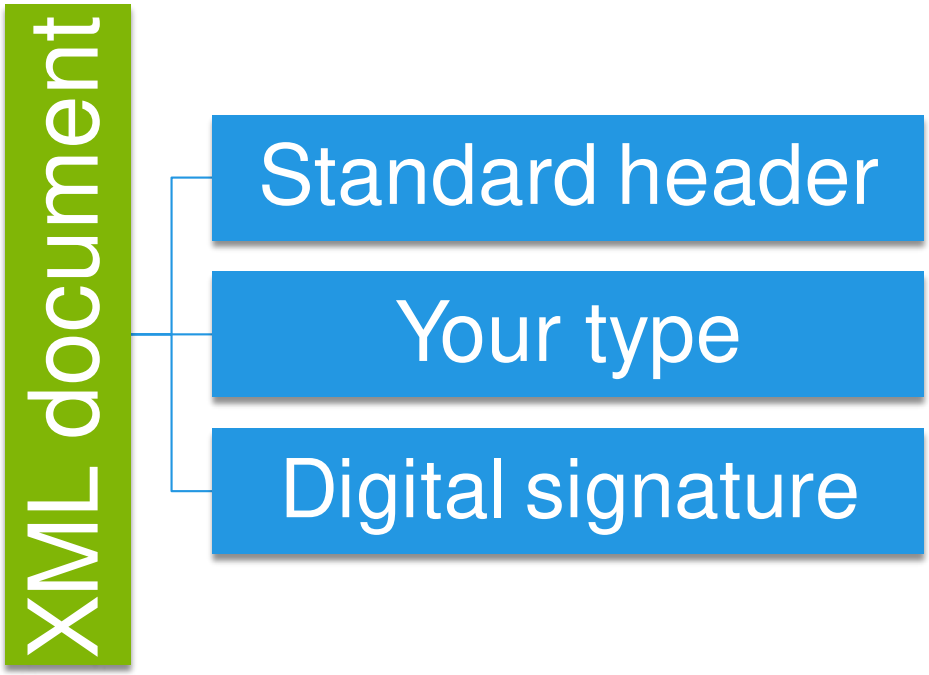
- Data types (static / dynamic typing)
- Definition of namespace
- Data that describes itself
- Rich set of tools and libraries

XML: Reuse of data types



- You don’t need to redefine “Date” type in order to define your own type

XML: Modular data types



- You can reuse existing standard for document structures, digital signature, etc.

XML message format

- <?xml version = "1.0"?>
- <tag attribute="value">
  <another-tag another-attribute="value" />
- </tag attribute="value">

Management of namespace through XML Namespace


```
<?xml version='1.0' encoding='UTF-8'?>
<soap:Envelope xmlns:soap='http://schemas.xmlsoap.org/soap/envelope/'
xmlns:xsi='http://www.w3.org/1999/XMLSchema-instance'
xmlns:xsd='http://www.w3.org/1999/XMLSchema'
xmlns:soapenc='http://schemas.xmlsoap.org/soap/encoding/'
soap:encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'>
 <soap:Body>
  <n:getQuoteResponse xmlns:n='urn:xmethods-delayed-quotes'>
  <Result xsi:type='xsd:float'>7.92</Result>
  </n:getQuoteResponse>
 </soap:Body>
</soap:Envelope>
```


Describing data types in XML: XML Schema


```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/1999/XMLSchema“
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
 <SOAP-ENV:Body>
  <namesp1:getQuote xmlns:namesp1="urn:xmethods-delayed-quotes">
  <symbol xsi:type="xsd:string">AKAM</symbol>
  </namesp1:getQuote>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```


XML: other features

- Self-descriptive: XML Schema
- Transformation tools: XPath, XSLT, etc.
- Rich set of API: DOM, SAX, etc.

- Message integrity and confidentiality:
  - XML Digital Signature
  - XML Encryption

JSON

- RFC 4627: The application/json Media Type for JavaScript Object Notation
- A text format for the serialization of structured data
- Four primitive types: string, number, boolean, null
- Two structured types: object, array

```
JSON object:
   {
      "Image": {
          "Width":  800,
          "Height": 600,
          "Title":  "View from 15th Floor",
          "Thumbnail": {
              "Url":    "http://bit.ly/",
              "Height": 125,
              "Width":  "100"
          },
          "IDs": [116, 943, 234, 38793]
      }
   }
```

JSON array

Presentation layer summary: a birds-eye view

Approach	Fixed	Extensible	
	Statically typed		Dynamically typed
binary	Packed binary	TLV	ASN.1
string		ASCII	XML, JSON

Q&A

Summary: Application layer protocols

- How do I interact with the other end?
  - You can roll your own protocol, but be cautious.
- How do I find the other end?
  - Three representative patterns
- How do I send information?
  - Presentation layer

