

広域分散アプリケーション特論

2005前期 月曜 3時限

場所: 情報基盤センター3F 多目的講義室

担当 青柳 睦

aoyagi@cc.kyushu-u.ac.jp

5月23日(月)

講義の内容, 成績評価方針(server-500.cc.kyu...)

サイエンスGrid NAREGI

Globus, Unicoreの現状

共通基盤層: セキュリティー CA, RA



講義の内容

■ グリッドの概要

- Gridコンピューティングとは
 - サイエンス分野での利用
 - ビジネス分野での利用

■ 計算科学の概論

- 主要なシミュレーション手法

■ サイエンスGrid NAREGI

- Globus, Unicoreの現状
- NAREGIミドルウェア概要
- 連成計算とその類型化

■ テーマ 考え中

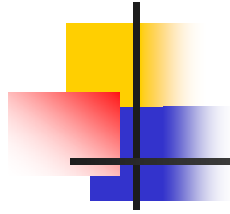
GT4の動向に依存・・・

講義資料はWebで公開
server-500.cc.kyushu-u.ac.jp



Globus Toolkit

- グリッドシステムの構築を支援するためのツールキット, 共通層, 他のミドルも必要
- グリッドに要求される様々な機能を実装、利用するためのユーザコマンド&ライブラリ群を提供
 - 分散された複数のコンピュータ上にサーバプロセスを配置
 - クライアントツールからこれらのサーバに接続
 - プログラムの実行やデータアクセスを可能にする基盤
- 米国 アルゴンヌ国立研究所と南カリフォルニア大学の共同開発
- デファクトスタンダードとして広く普及
 - 世界中のほとんどすべてのグリッドプロジェクトで採用されている



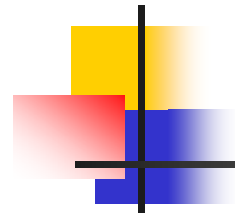
Globus Toolkit

- 3つのバージョンが並存
 - バージョン2系 – 2.4.3 がstable release
 - HTTP,LDAPなどをベースとする独自プロトコル
 - バージョン3系 – 3.2.1 がstable release
 - グリッドサービスを採用
 - バージョン4系 – 3.9.2 (development release) ⇒ 4.0.0 (2005.4.29 正式リリース)
 - WSRF対応
 - <http://www.globus.org/toolkit/>



Webサービスとグリッドサービス

機構	Webサービス	グリッドサービス
オブジェクト参照	なし	GSHとGSRおよびMapper
インターフェイスの表現	WSDL	WSDL + 拡張
メッセージパッシング	SOAP	SOAP (その他の実装も可能)
オブジェクトの検索	UDDI	Service Group サービス
オブジェクトの生成	なし	Factory サービス
オブジェクトの破棄	なし	明示的なoperation もしくはソフトステイト
オブジェクトの内部状態	なし	グリッドサービスデータ
通知	なし	Notification サービス

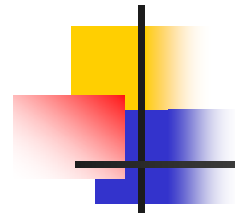


GT2, 3, 4比較 (1) OGSA以前

GT2

GGF4@トロント 2002.2 以前





GT2, 3, 4比較 (2) OGSA以降

GGF4@トロント 2002.2 以降 GlobusWorld@サンフランシスコ
2004.1以前

GT3

アプリケーション

アプリケーション

ミドルウェア

OGSAサービス群

他ミドルウェア

基盤ソフト

GT3

プロトコル

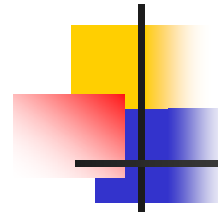
SOAP / GWSDDL

計算資源、ネットワーク資源



Globus Toolkit 3とOGSI

- Globus Toolkit 2
 - グリッドの下位レベルインフラのデファクトスタンダード
 - バージョン2までは独自プロトコルを使用
- Globus Toolkit 3
 - OGSIのサンプル実装
 - OGSIを用いて実装した次世代のGlobus Toolkit
 - 基本的なコンセプトはGT2までと同じ
 - プロキシ証明書を用いたdelegate可能な認証システム
 - Mapfileを用いたUnixユーザへのマッピング
 - クライアントはJavaとC++
 - サーバ側はJavaで実装



- **WSRF: Web Service Resource Framework**
- **OGSIはなかったことに・・・**
 - OGSAはWSRF上に実装
- **OGSIを6つのWeb Service コンポーネントに分割し、リファクタリングしたWSRFに再構成**
- **WSRFがOGSIを完全にリプレイス**
- **WSRFをベースにしたGT4が登場**
 - 今後はGT3はお払い箱？

GT2, 3, 4比較 (3) WSRF以降

GT4

アプリケーション

アプリケーション

ミドルウェア

OGSAサービス群

他ミドルウェア

基盤ソフト

プロトコル

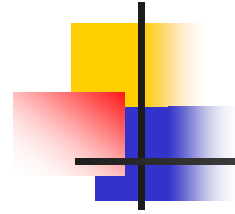
Web-Service including WSRF

計算資源、ネットワーク資源



OGSIとWSRFの対応

OGSI	WSRF
Grid Service Reference	<i>WS-Addressing</i> Endpoint Reference
Grid Service Handle	<i>WS-Addressing</i> Endpoint Reference
HandleResolver ポートタイプ	WS-RenewableReferences
サービスデータ	WS-ResourceProperties
GridService 生存期間管理	WS-ResourceLifeCycle
Notification ポートタイプ	WS-Notification
Factory ポートタイプ	Treated as a pattern
ServiceGroupポートタイプ	WS-ServiceGroup
Base fault type	WS-BaseFaults



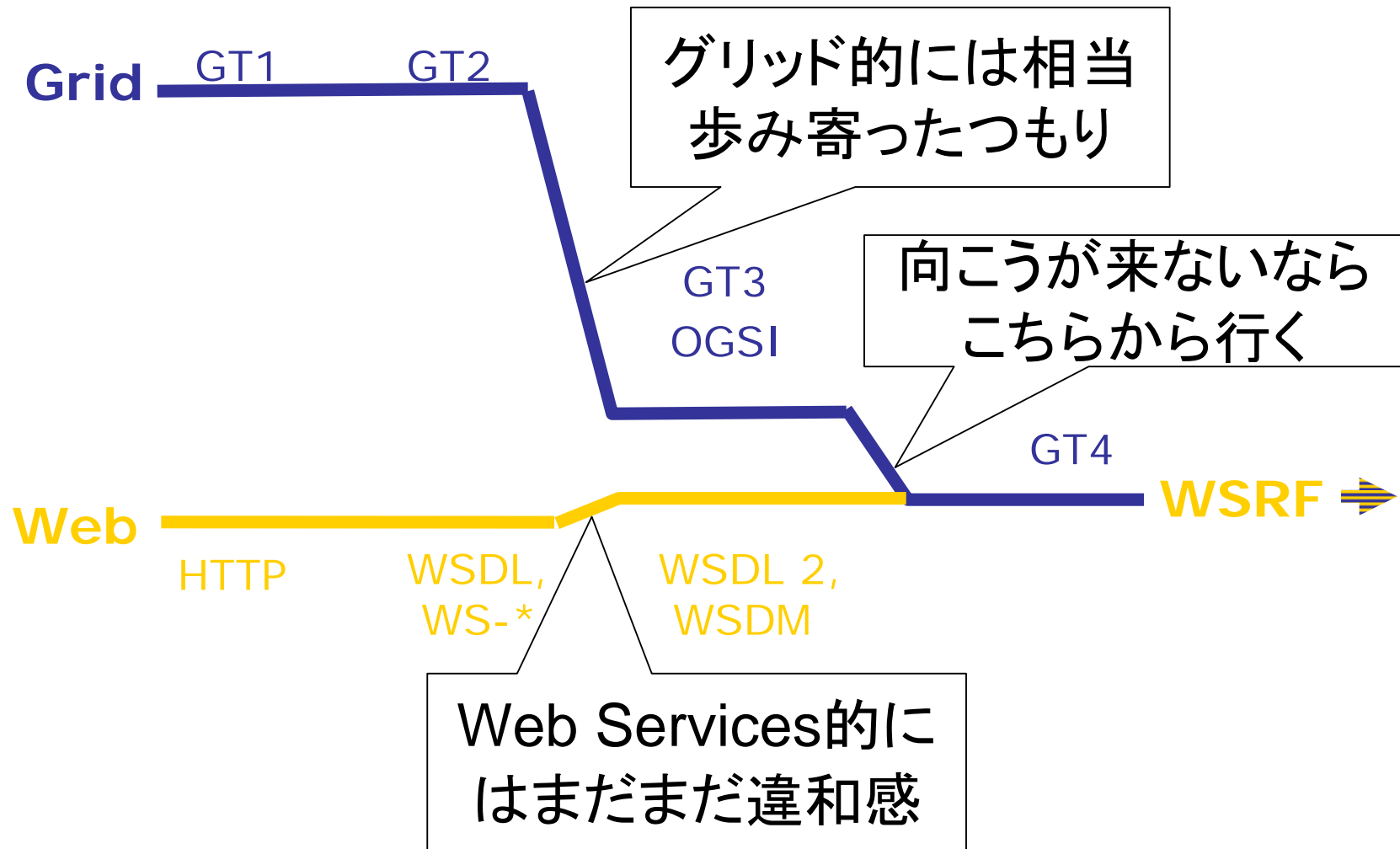
WSRF 標準化

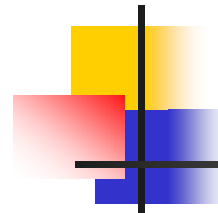
- OASISで標準化

- Web Services Resource Framework と Web Services Notificationの2つのTC (Technical Committee)が設立。
- Web Services Resource Framework
 - David Snelling(Fujitsu), Ian Robinson(IBM)
- Web Services Notification
 - Willam Vambenepe(HP)

GridとWSRF – 本当は？

産総研: 田中氏より





GT2,3,4 共通の基盤

- シングルサインオン認証
 - GSI
- 資源情報の取得
 - MDS
- リモートサーバ上でのプロセス実行
 - GRAM
- データ転送
 - GridFTP



シングルサインオン認証 Globusのセキュリティ基盤(GSI)

- 公開鍵を用いた認証と認可
 - 認証: 相手が誰であるかがわかること
 - SSL をベース
 - 認可: わかった相手に特定の操作を許可すること
- シングルサインオン
 - 一度の操作でグリッド全体に対してログインできること
 - プロキシ証明書で実現

Symmetric Cryptography(共通鍵暗号方式)(intro.)

■ 単純な暗号化:

■ 例)

My name is James Bond.

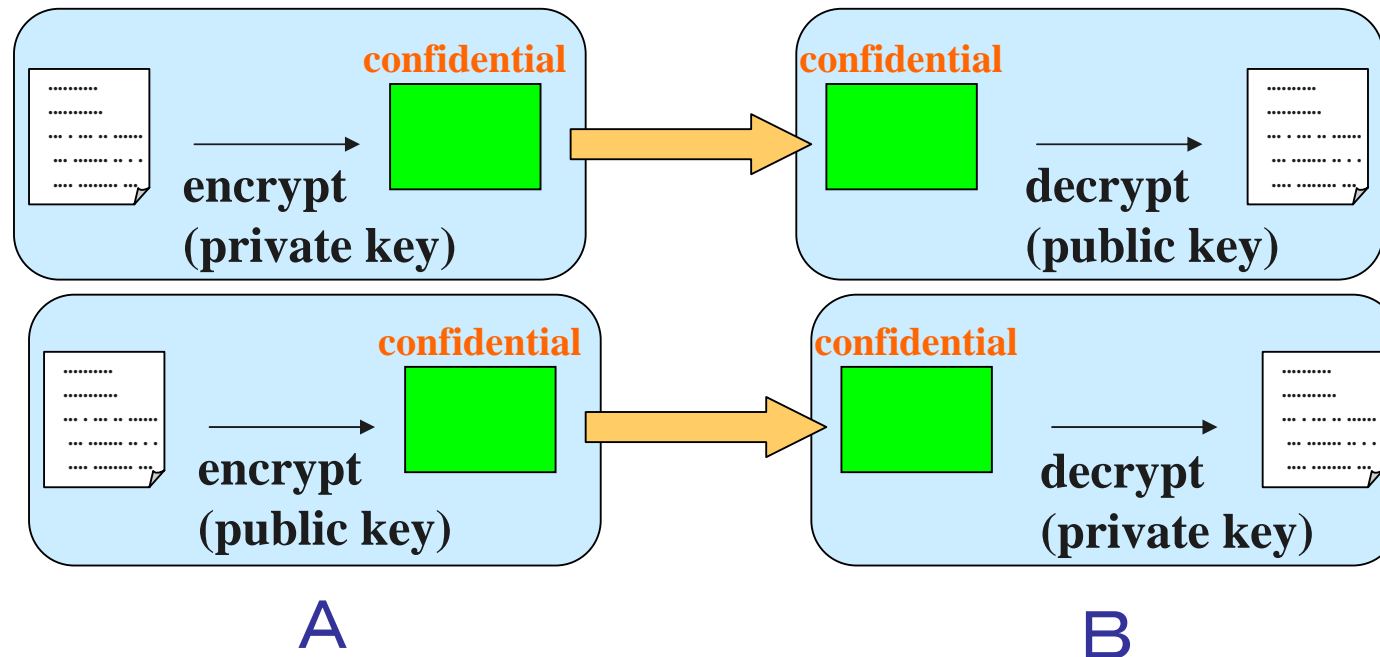
↓ shift 2

Oa pcog ku Lcogu Dqpf.

- encrypt(暗号化), decrypt(復号) に同じ鍵(=2)を利用.
- 実際にはもっと複雑なアルゴリズム (DES, AES, IDEA, etc.)
- 不特定多数と通信する際の問題点
 - 鍵を安全に相手に渡す方法
 - 相手に応じて異なる鍵を用意

public key cryptography 公開鍵暗号方式

- 鍵の対(private key, public key)を使って encrypt



- Point: encrypt に用いた鍵では decrypt 出来ない.
 - public key を公開し, private key は隠す.
 - アルゴリズム: RSA, Rabin, Diffie-Hellman, etc.



symmetric key と public key の併用

- symmetric key cryptography
 - 安全に key を相手に送る手段が必要
- public key cryptography
 - encrypt, decrypt が遅い(symmetric key の数百～数千倍)ので大きいサイズの通信に不向き

- 両方式の併用：
 - Authentication(認証)と symmetric key の送信に public key cryptography 利用.
 - データの送受信には symmetric key crypt. を利用.



Authentication(認証)

- private key の owner の確認
 - ある public key について, 対応する private key の owner は, 本当に本人か?
 - 他人による impersonation (成りすまし) の防止
- 手段:
 - 例えば, 本人と確認できるものの提示:
運転免許証, 学生証等
 - 毎回やるのは面倒
 - インターネット上では非現実的 なので・・・CA



Certificate (証明書)

- 信頼できる authority(機関)が発行する本人の証明書
 - 発行時に十分な本人確認.
 - private key, public key の対について,
「private key の owner」と
「public key」 の関係を保証.
 - インターネット上で送受信できる電子的情報として発行.
 - 発行機関の digital signature により内容保証

(c. f.)類似の認証手段: パスポート 外務省の印鑑

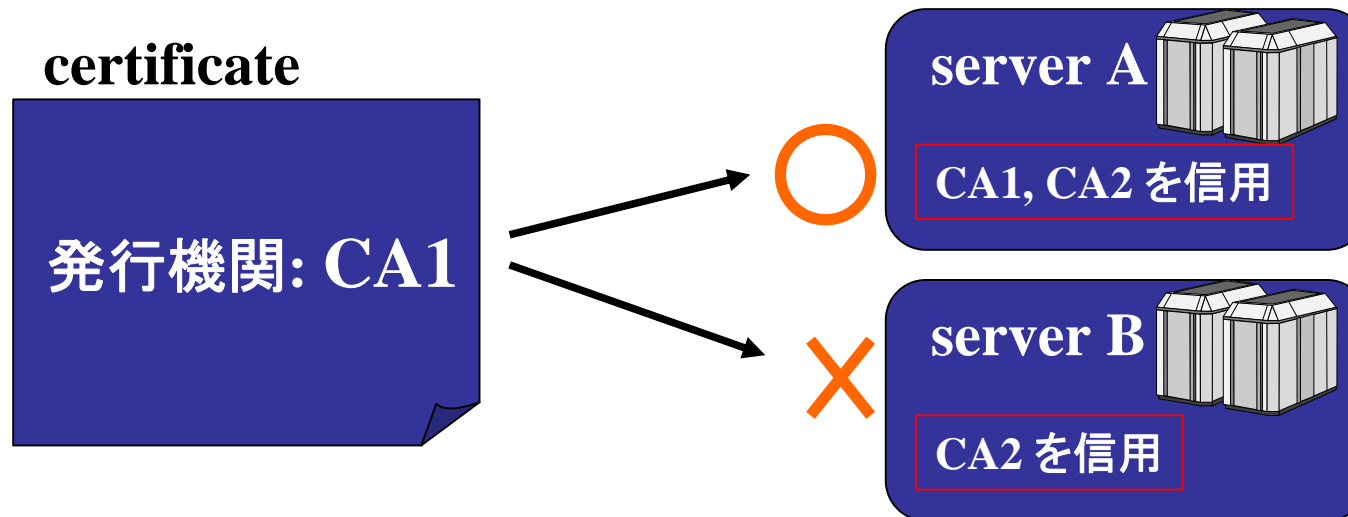


Registration Authority(登録局)

- PKIにおいて,本人確認を行う機関.
 - 通常,利用者に近い場所に設置
 - 本人確認の手段:
 - 身分証明書(運転免許,パスポート等) + 対面
 - 遠い場合は電子メール等
- 確認できたら,発行機関に証明書発行依頼
 - 本人確認は時間がかかるので
大きな組織では発行機関(CA)と別に用意
 - 必要に応じて複数
例) 各学部にはRAを設置し,大学事務局で証明書発行
 - 小さな組織では発行機関が RA を兼務

Certification Authority(認証局)

- 証明書の発行機関
 - private key の所有者と public key の関係を保証.
 - CA の digital signature を添付.
- どの CA を信用するかはserverが選択.



certificate

以下の内容のテキストファイル

Certificate:

...

Issuer: C=US, O=Globus, CN=Globus Certification Authority

Validity

Not Before: Feb 26 13:24:16 2002 GMT

Not After : Feb 26 13:24:16 2003 GMT

Subject: O=Grid, O=Globus, OU=cc.kyushu-u.ac.jp, CN=Takeshi Nanri

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public Key: (1024 bit)

Modulus (1024 bit):

00:b3:06:66:93:f4:54:16:21:d0:3c:15:3f:0e:f8:

...

a7:f2:fb:7f:07:68:96:4d:f7

Exponent: 65537 (0x10001)

X509v3 extensions:

Netscape Cert Type:

SSL Client, SSL Server

Signature Algorithm: md5WithRSAEncryption

25:8b:38:4e:0d:57:e8:73:97:b8:30:1c:10:ac:c2:15:ba:b4:

...

5e:9b

CA

validity

Distinguished Name
(識別名)

利用者のpublic key

CAの
digital signature

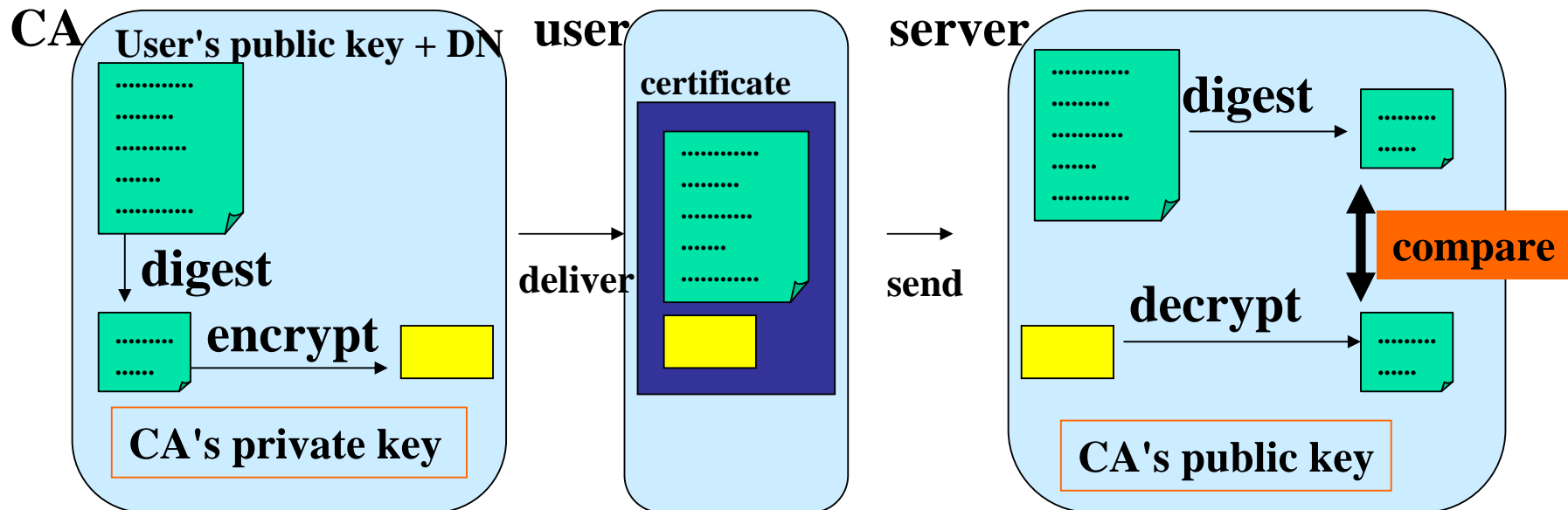
Distinguished Name



- PKI 上で個人(や特定の計算機など)を識別するための名前
 - 通常, いくつかの情報の組み合わせ
 - country
 - locality
 - organization
 - section
 - common name(固有名詞)
 - E-mail アドレス
 - 一つのCA内で重複しない.

Digital Signature

- 証明内容を CA の private key で encrypt
 - CA の public key で decrypt することにより内容証明
 - 証明書全体のencrypt は時間がかかるので証明書の digest(要約) を利用
 - 各サーバは信用する CA の public key を保持

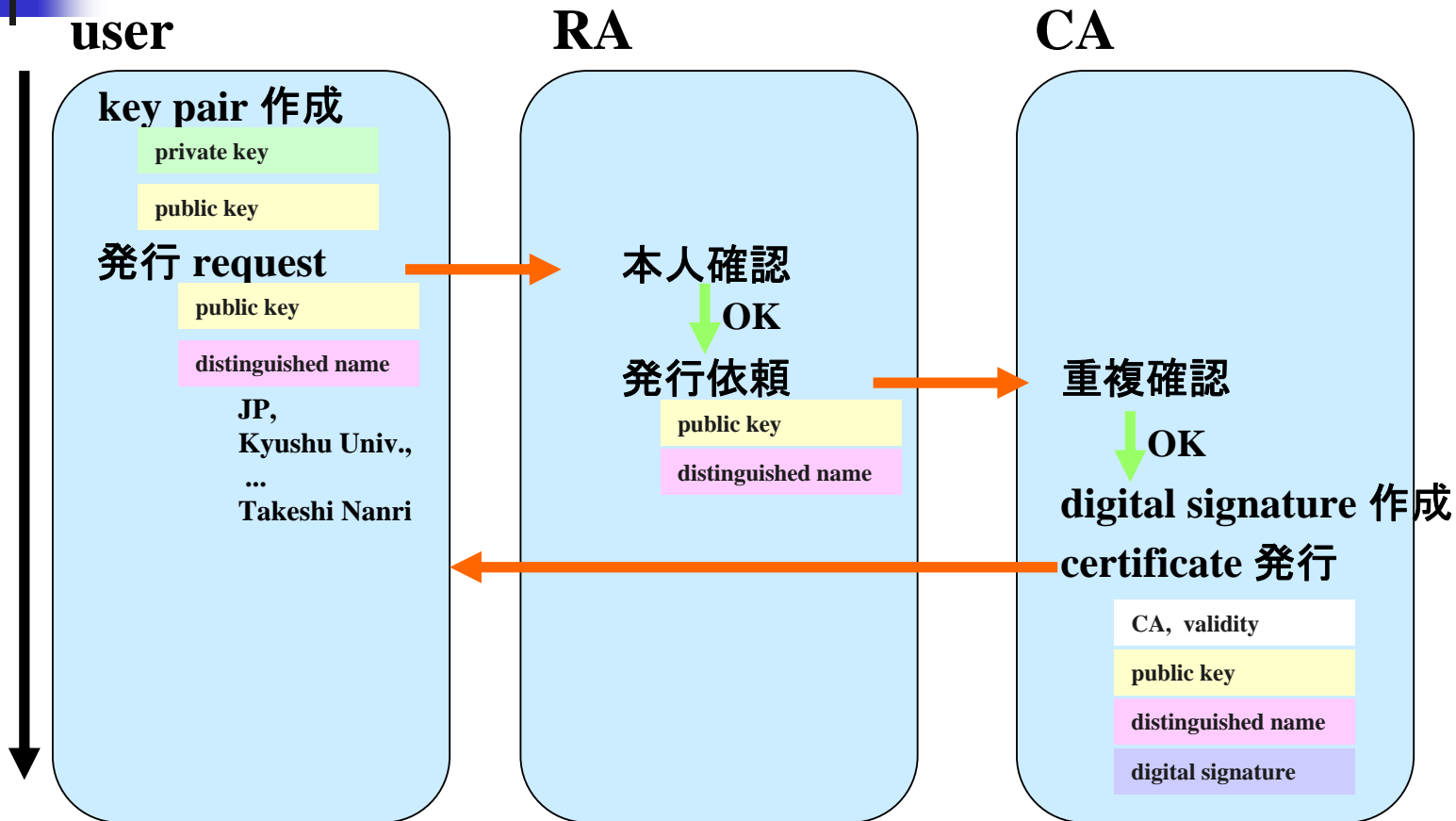




Message Digest

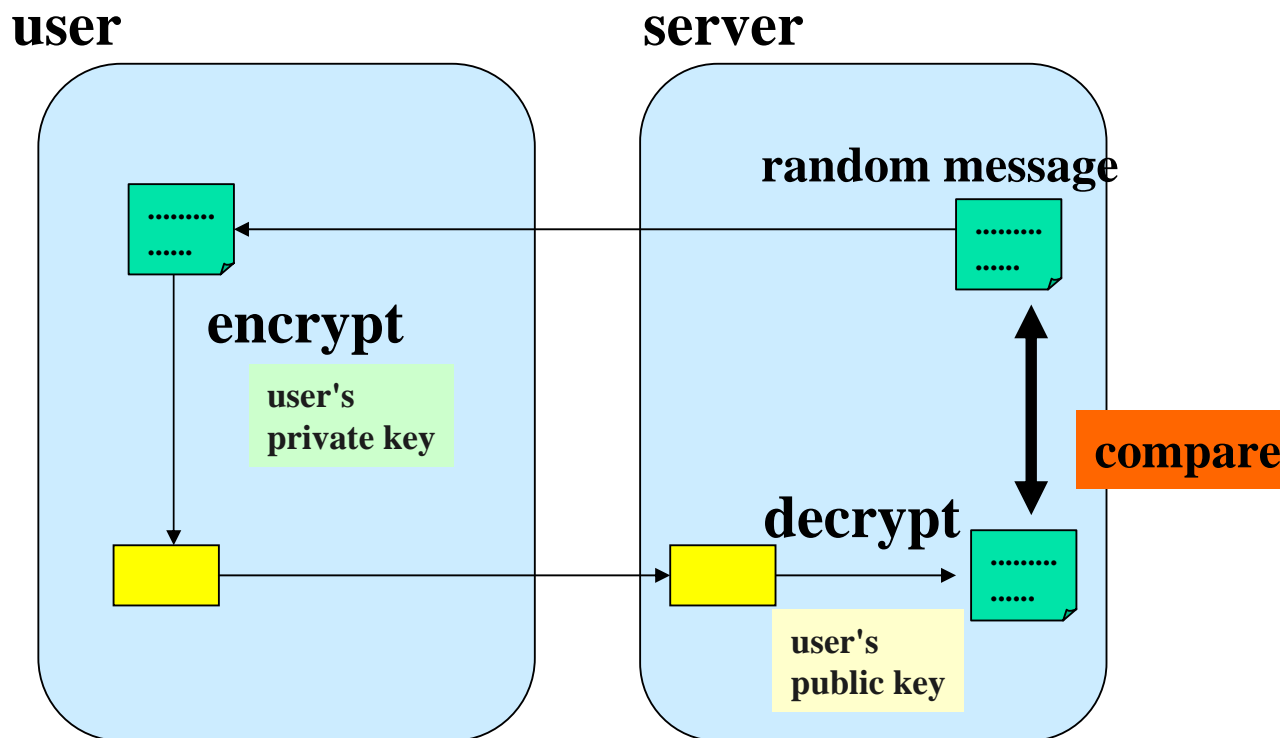
- 長いメッセージから固定バイト長(20byte程度)の digest を作成.
 - public key crypt. による encrypt の時間短縮
- 改ざんの防止
 - public key で decrypt 可能なので,
「同じ digest が生成されるようにメッセージを
書き換える」ことが出来ると digital signature の
信頼性が無くなる.
 - 特殊な関数:
 - digest から元の文への復元は不可能
 - 元の文が 1bit でも違うと digest は大きく変化
 - アルゴリズム: MD5, SHA1, etc.

certificate 発行までの流れ



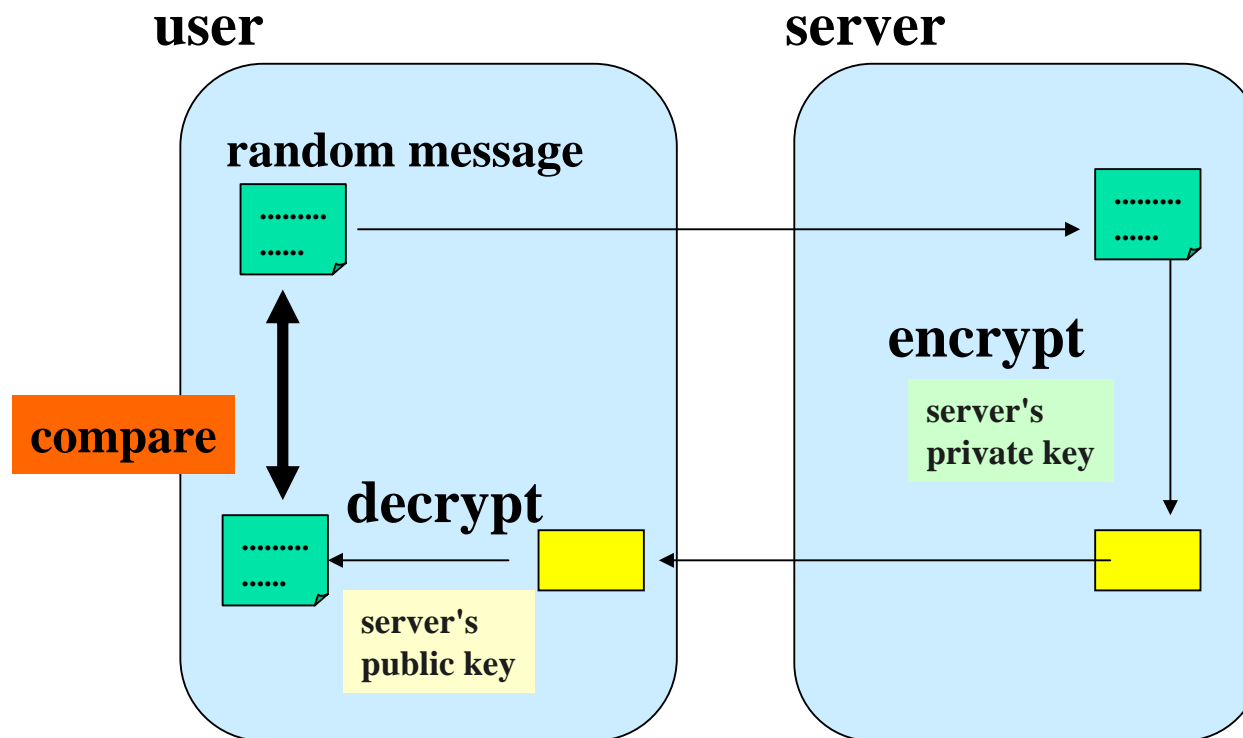
ユーザの Authentication

- public key と本人の関係が保証されているので、public key cryptography を用いた確認可能



サーバのAuthentication

- サーバの impersonation (成りすまし)による問題:
 - 想定していたサーバ以外にジョブやデータを送ることによる情報の漏洩等





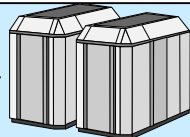
Authorization (認可)

- Authenticate された利用者に対し, 適切な権利を与える.
 - Grid の場合, 適切なユーザIDへの mapping (対応付け).
 - Globus : Grid-Mapfile

ユーザ ID への mapping

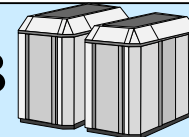
- 利用者を適切なユーザIDに割り当てる
- データベースによる管理:
 - Distinguished Name とサーバ上のユーザIDの mapping

server A



Takeshi Nanri, Kyushu Univ., Japan
→ nanri

server B




Takeshi Nanri, Kyushu Univ., Japan
→ guest



private key の扱い

- 公開鍵方式では private key の管理が最も重要
 - 盗まれると簡単に"成りすまし"が可能.
- private key を隠す.
 - 残念ながら, 計算機の管理者が信用できない場合もありうるので, private key をさらに暗号化して格納.
 - private key を参照するにはパスワードが必要

サーバにおける本人確認のため毎回 private key で暗号化
→ 毎回パスワードを入力するのは面倒⇒代理認証方式



GTにおけるSingle sign-on

- 一度パスワードを入力すると、以降は自動的に認証する仕組み
 - 複数の計算機を seamless に利用する
Grid 環境では特に有効
- Globus Toolkit では, proxy(代理人)を利用

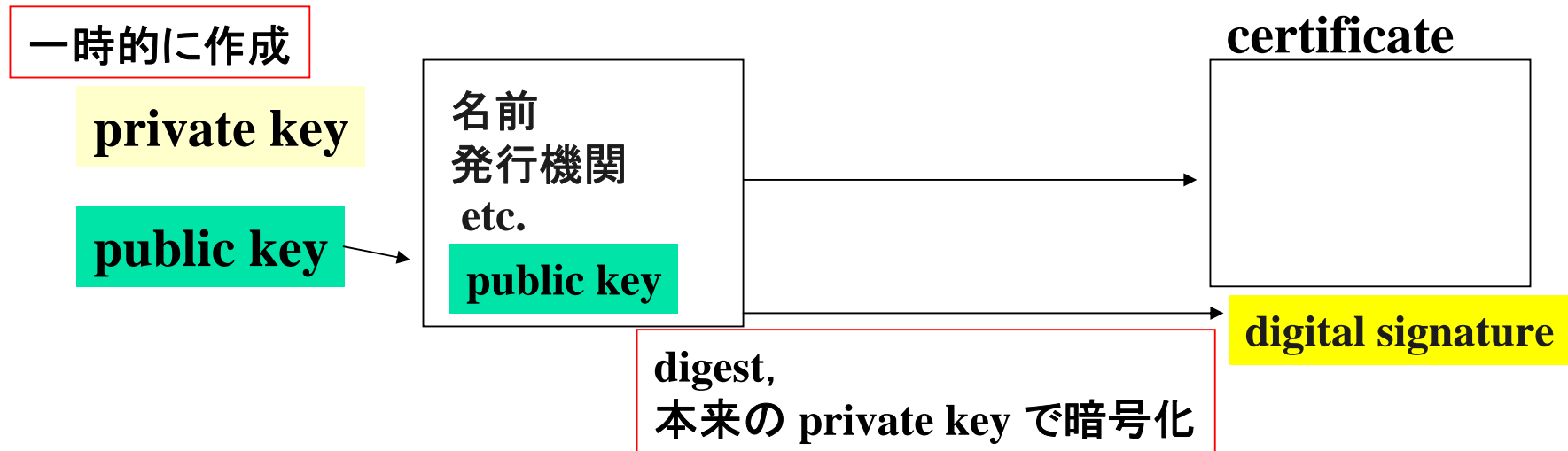


proxy

- 一時的に利用可能な public key と private key を持った proxy (代理人) を作成.
 - Globus Toolkit では, 特に指定しなければ 12時間.
- digital signatureにより, 利用者の代理人であることを保証.
- 有効時間が比較的短いため, private key を暗号化する必要なし.
 - 認証の度にパスワードを入力しなくて良い.

proxyの作成

- 短期間利用可能なprivate key と public key の対を作成.
- それらを元に certificate を作成.
- certificate に本来の private key で電子署名.
 - 一度だけパスワードの入力が必要.



proxy の certificate

Certificate:

Data:

...

Issuer: O=Grid, O=Globus, OU=cc.kyushu-u.ac.jp, CN=Takeshi Nanri

Validity

Not Before: Dec 1 04:42:12 2002 GMT

Not After : Dec 1 16:47:12 2002 GMT

Subject: O=Grid, O=Globus, OU=cc.kyushu-u.ac.jp, CN=Takeshi Nanri, CN=proxy

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public Key: (512 bit)

Modulus (512 bit):

00:ac:f1:8f:81:98:04:ef:da:6a:a1:53:4e:53:ea:

...

4e:1b:4b:7f:e7

Exponent: 65537 (0x10001)

Signature Algorithm: md5WithRSAEncryption

18:7a:a4:9c:7c:50:89:9f:97:e5:55:8b:aa:2a:a2:ae:f7:a3:

...

b4:23

CA = user

validity = 12h

proxy's DN

temporal public key

user's
digital signature

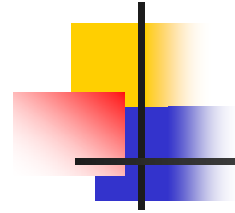


proxy による認証（二段階方式）

- 2つの certificate を利用
 - ユーザの certificate
 - proxy の certificate
- proxy が正規の代理人であることを確認
 - CA のpublic keyでユーザの certificate の digital signature を確認
 - ユーザの certificate に含まれる本人の public key で proxy の certificate の digital signature を確認

GlobusにおけるSingle SignOn (続)

- 事前にいずれかのCAから証明書を発行してもらう
 - Globus でもCAを運用している
 - 他のCAを使用する場合にはそのCAの証明書をインストールする必要がある
- 使用する際にgrid-proxy-init を実行
 - Globusによるグリッドにログインするためのコマンド:
ここで自分の証明書のパスフレーズを入力
 - パスフレーズなしで使用できるプロキシ証明書を作成してテンポラリディレクトリに書き込んでいる



次週・・GlobusのMDSから続き
の予定です