

CAS を利用した Single Sign On 環境の構築

内藤 久資

(Hisashi NAITO)

`naito@math.nagoya-u.ac.jp`

名古屋大学多元数理科学研究科

Graduate School of Mathematics, Nagoya University

Plan of Talk

- CAS および CAS² の簡単な紹介
- CAS を利用した Single Sign On 環境の実例の紹介
名古屋大学での CAS の利用状況
- CAS 及び CAS² の仕組み
- CAS の利点
- 簡単なデモ

CAS & CAS2 とは

- CAS : Yale 大学が開発した Open Source software
 - Web Application のための Authentication 環境
 - 現在は JA-SIG の Official Project
 - 強力な “Authorization” 機能を追加 (CAS²)
- 特徴
 - Web Application に対する認証に特化したシステム
 - Single Sign On 環境を容易に実現可能
 - Web Application 側には特権が必要ない
 - Kerberos に似た Authentication 環境

名古屋大学での利用状況

- 「名古屋大学ポータル」で CAS² サーバを設置
 - 「全学 ID」 LDAP サーバ（情報連携基盤センター）
- CAS² を利用した Web Application 群
 - 名古屋大学ポータル（情報連携基盤センター）
 - 新教務システム（学務情報掛）
 - 法学部教育支援システム
 - 全学共通教育シラバスシステム（高等教育センター）
 - 核燃料管理システム（工学部）
 - Web CT（情報メディア教育センター）
 - 研究者プロフィール（ ??? ）
- これらのアプリケーション群が Single Sign On で利用可能

CAS CAS2 認証のしくみ

● 用意すべきもの

- Web Application Server (including CAS client)
- CAS Server (over Tomcat)
- Directory Server (example LDAP Server)
- Web Browser (Client, User)

● 登場する概念

- Ticket Granting Cookie (TGC)
 - ユーザが認証済みかどうかを判断する
 - Kerberos の “ticket granting ticket” に相当する概念
- Service Ticket (ST)
 - ユーザが Web Application にアクセスするための One Time Ticket

CAS CAS2 認証のしくみ

- Login しているユーザのブラウザには TGC が保存される
 - CAS Server は TGC & ST データベースを保存
- 1 回のアクセスごとに ST を発行
 - ST は One Time Ticket
 - ST は TGC に付随
- TGC で Authentication, **ST で Authorization を行う**
- ST Validation Application に「ユーザのデータ」を送信
- TGC の Timeout = Session Timeout
- TGC の削除 = Logout

CAS2 の権限管理手法

- Authorization のためのデータベース (CAS-ACL) を導入
 - どの Web Application に対して (target URL)
 - どのユーザが (User Information)
 - いつ (Access Time)
 - どこから (Client Information)アクセスできるかを記述したリスト
- CAS-ACL の各エントリごとに「Web Application に対し、どの User Information を返すか」を指定
- CAS-ACL の各エントリに属する URL を“Access Control Class” (CAS-ACC) と呼ぶ
- ST にはどの CAS-ACC に属するかの情報が付随している

CAS-ACC の Example

```
dn: cn=gakumu-kykr,ou=gakumu,ou=cas,o=nagoyaUniv
cas-allow: (&(uid=naito)(date>=20051010)
(date<=20051110)(IP=133.6.130.0/24))
cas-service: https://app.*\.myynu\.jp/kyoin/kykr.+
cas-attributes: uid,mailAddress,IdNo,FullName,dn
```

- 以下の条件の時, この CAS-ACC にマッチ
 - URL: `https://app.*\.myynu\.jp/kyoin/kykr.+`
 - ユーザ (uid) が `naito`
 - アクセス時刻が 2005/10/10 から 2005/11/10 の間
 - クライアントが `133.6.130.0/24` に属する
- このとき, Web Application に対して, User 情報として `uid,mailAddress,IdNo,FullName,dn` を送信する

CAS client module の組み込み

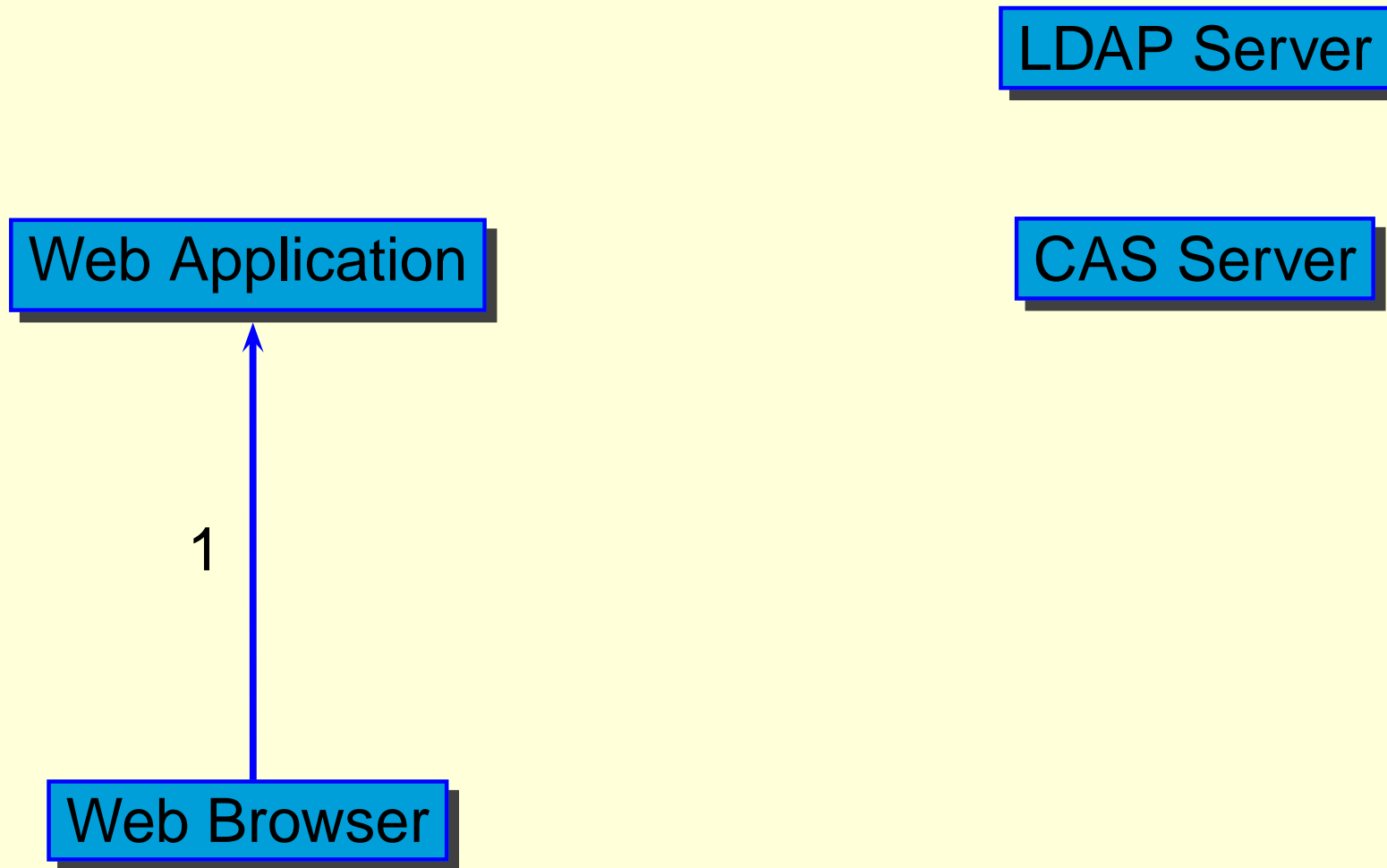
● Java Servlet の場合

```
private void doGet(
    HttpServletRequest request,
    HttpServletResponse response)
    throws ServletException, IOException {
    /******* initial setting *****/
    /******* Perform CAS Client *****/
    CasClient cas = new CasClient() ;
    cas.setServiceURL(CAS_SERVICE_URL) ;
    if (!cas.casPerform(request, response)) return ;
    Map r = cas.getResult() ;
    String xmlResponse = cas.getResponse() ;
    /******* End of Perform CAS Client *****/
    if (r != null) request.setAttribute("userid", r.get("uid")) ;
    app.getRequestDispatcher(loginForm).forward(request, response) ;
    return ;
}
```

CAS 認証のしくみ (1: Login)

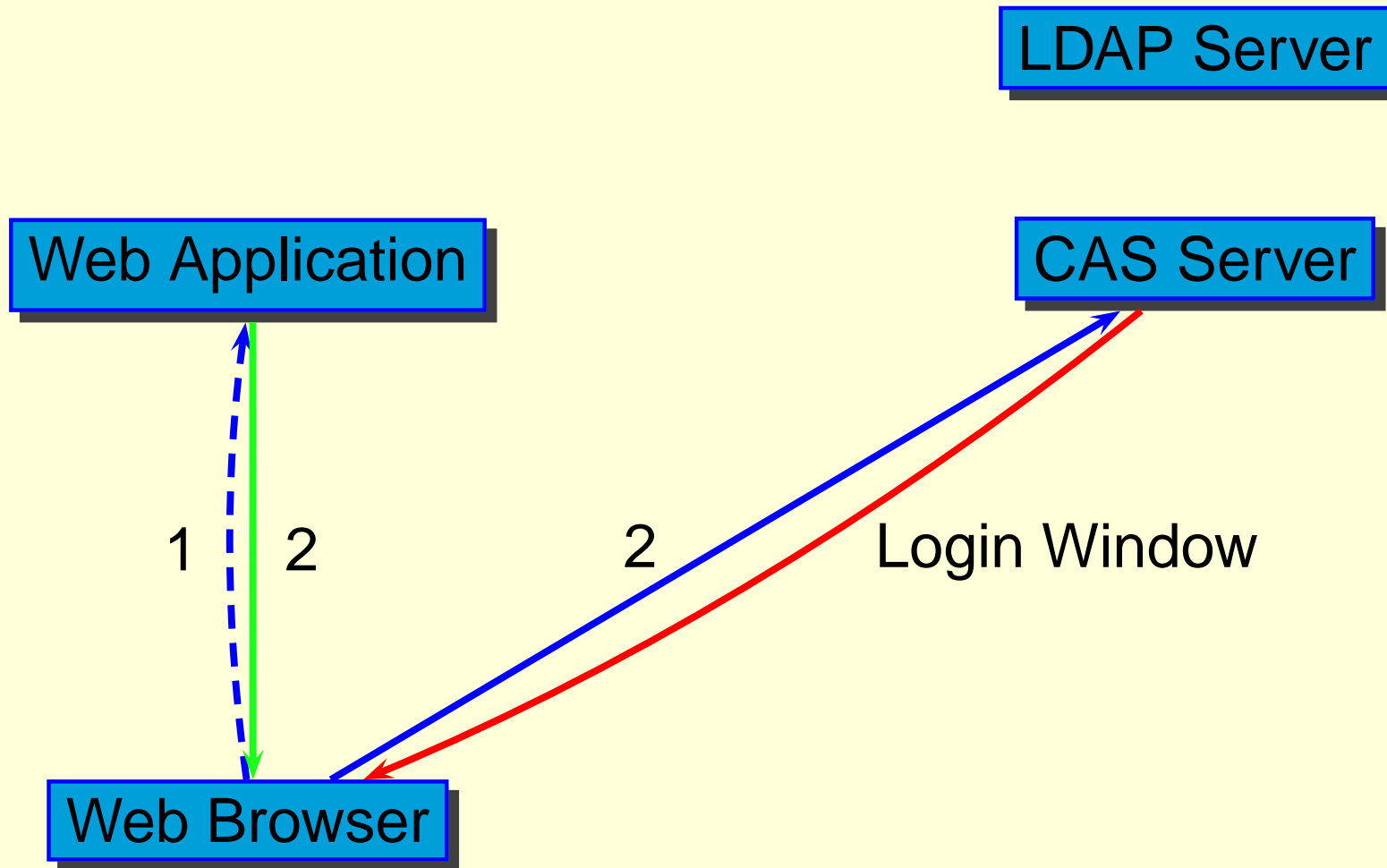
- ユーザ (Browser) が CAS² Application 群にはじめてアクセス
- Browser には CAS² の TGC が存在していない
- ユーザ認証を行う必要がある
- 以下の通信は **SSL** で暗号化されていることが必要
 - Browser \iff CAS Server
 - Web Application \iff CAS Server

CAS 認証のしくみ (1: Login (1))



1. Access to <https://aFQDN/a.html>

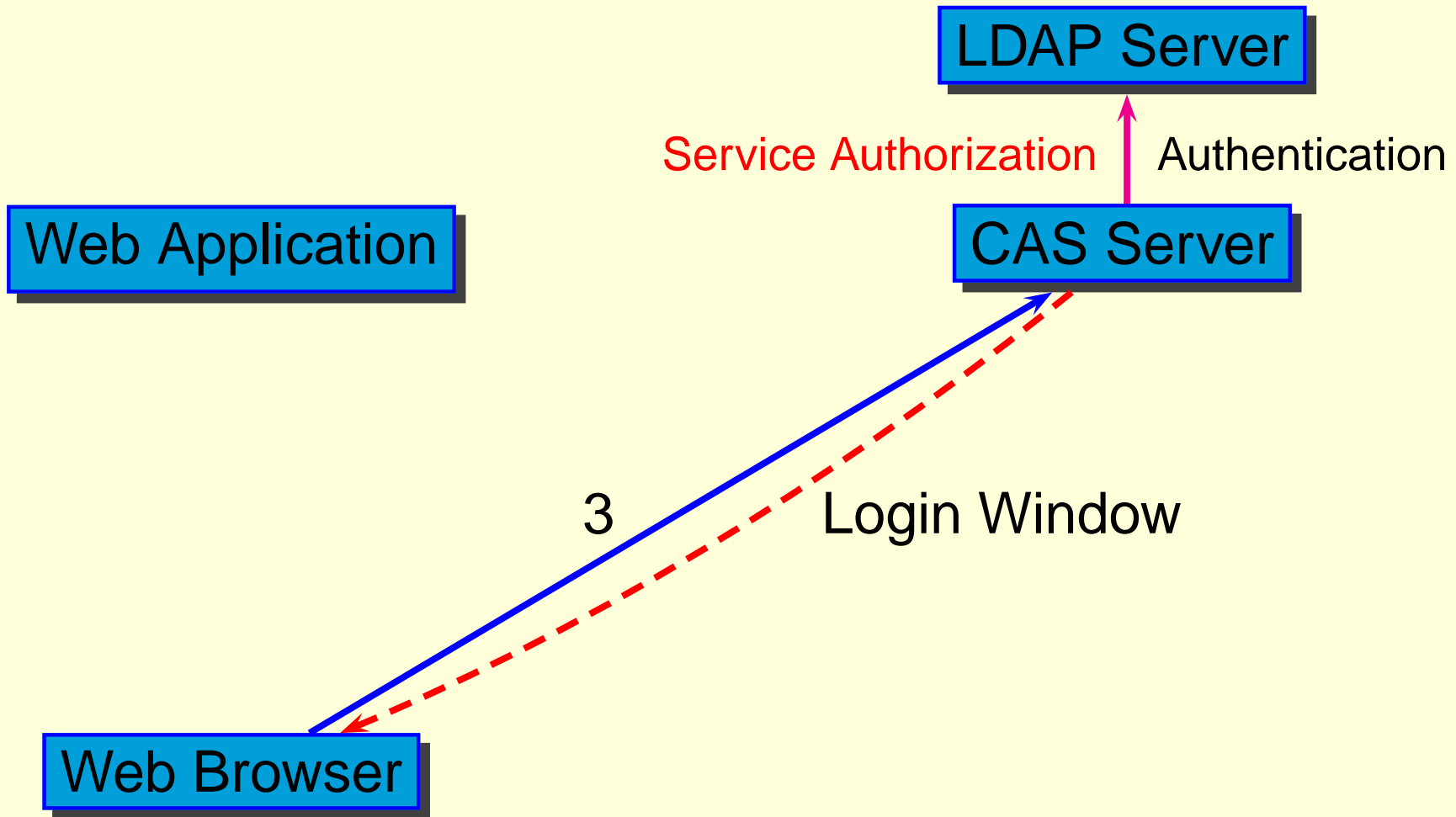
CAS 認証のしくみ (1: Login (2))



2. Redirect to

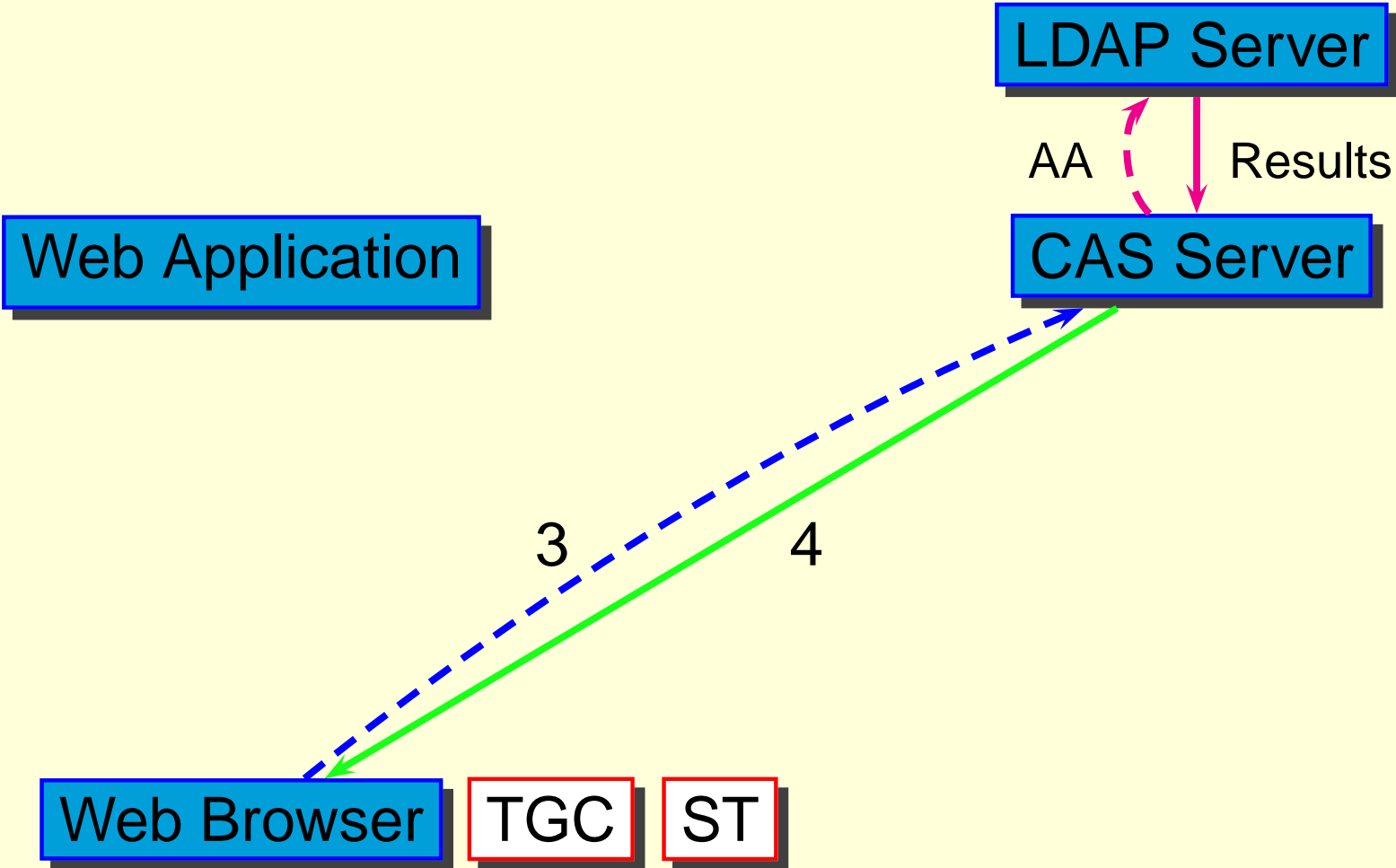
<https://CAS/login&service=https://aFQDN/a.html>

CAS 認証のしくみ (1: Login (3))



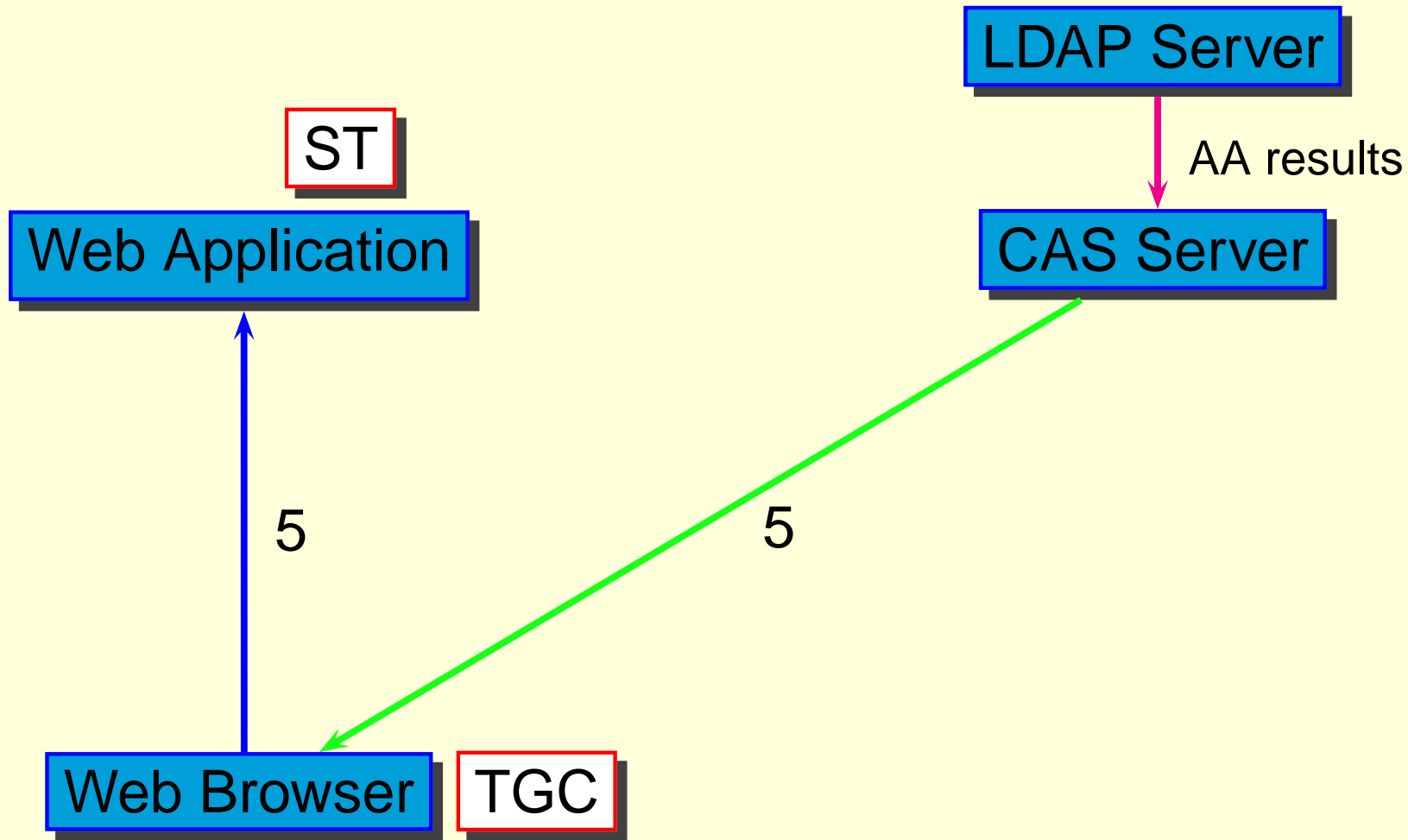
3. Input UserID & Password with service <https://aFQDN/a.html>

CAS 認証のしくみ (1: Login (4))



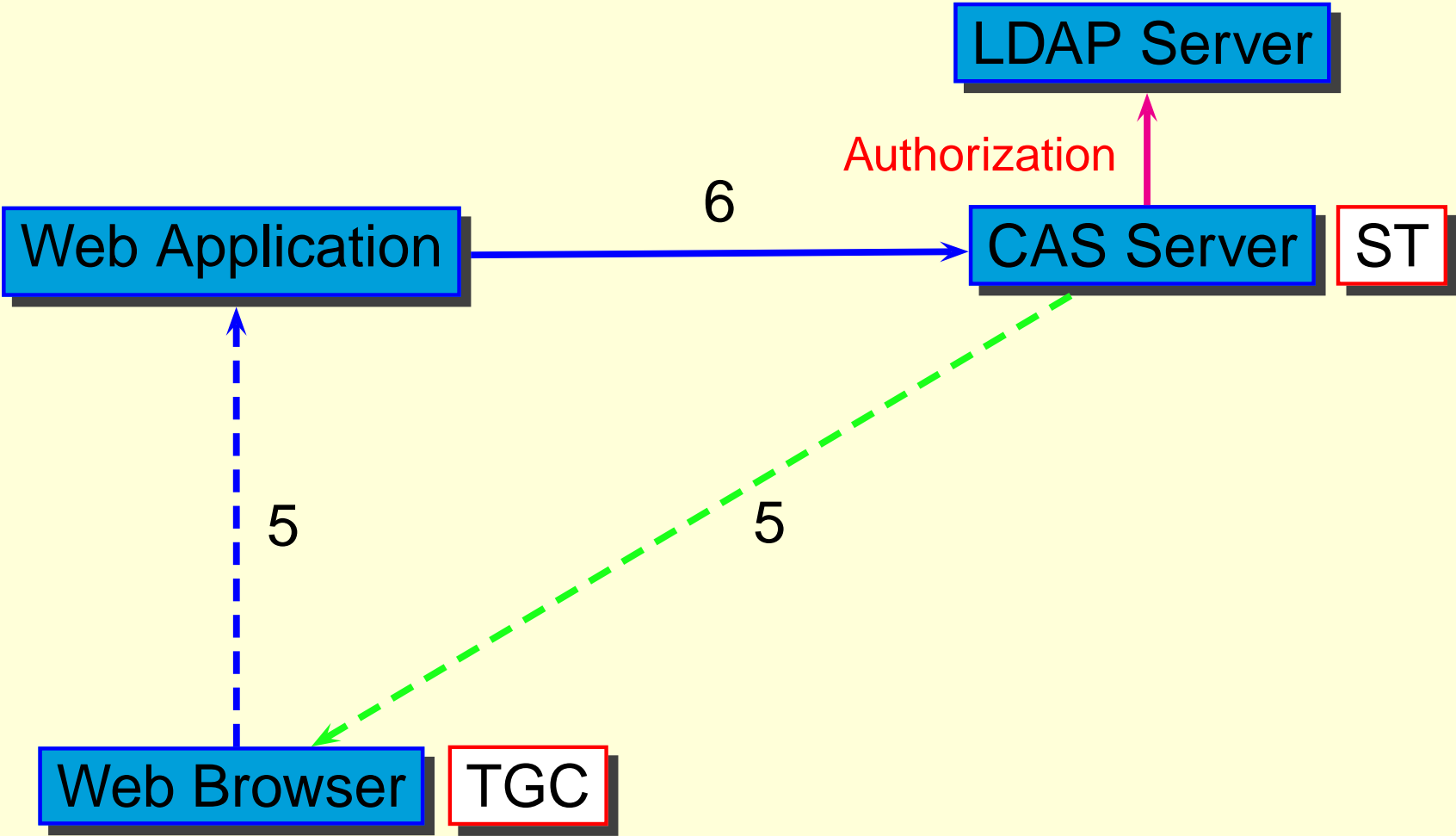
4. Send Ticket Granting Cookie to Browser

CAS 認証のしくみ (1: Login (5))



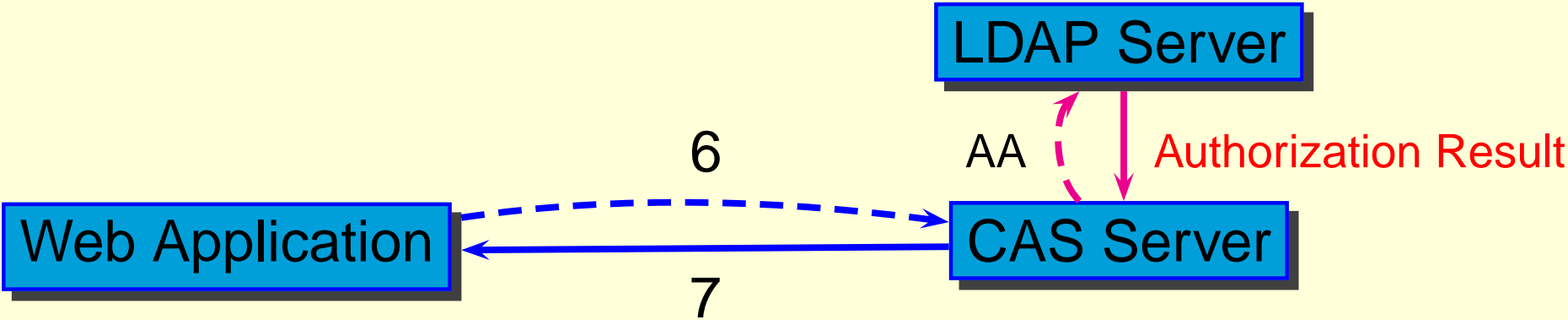
5. Redirect to <https://aFQDN/a.html&ticket=ST-xxx>

CAS 認証のしくみ (1: Login (6))



6. Verify Service Ticket

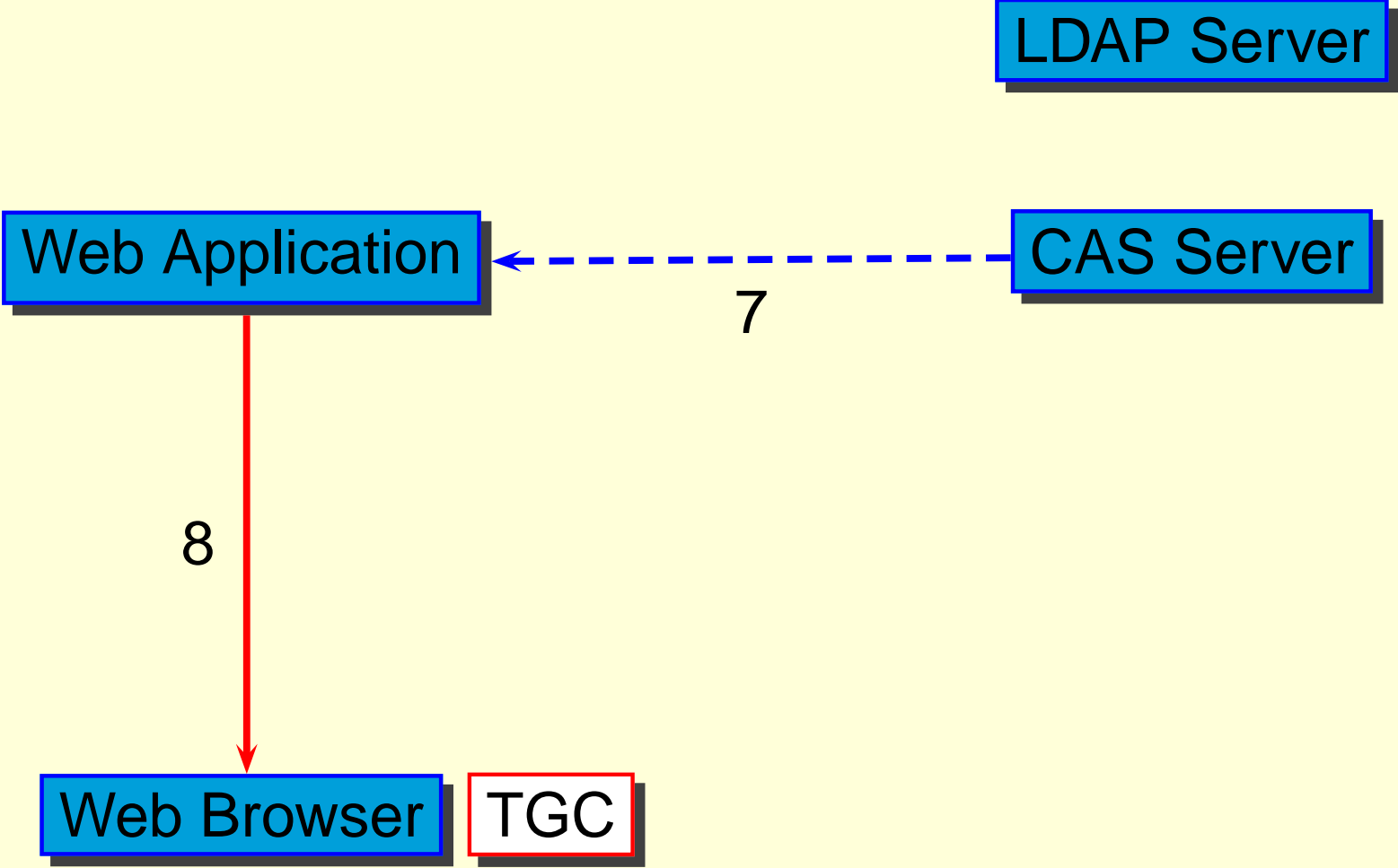
CAS 認証のしくみ (1: Login (7))



Web Browser TGC

7. Receive verify result form CAS server

CAS 認証のしくみ (1: Login (8))



8. Receive Data from Application Server

CAS 認証のしくみ (注意事項)

- Login の一連の操作
 - 一見すると何度もアクセスが発生している
 - JavaScript/HTTP redirection が多数を占める
- 実際にユーザから visible なアクセス : 以下の 2 回
 - Login Window
 - 実際のページの取得

CAS 認証のしくみ (2: Verify Ticket)

- Login 後のアクセス (Browser が有効な TGC を持っている)
 - アクセスごとに TGC の “count down timer” を更新
 - ST による Authorization
 - ST が属する CAS-ACC と異なるアクセス時
 - Login への redirection を発行
 - Authorization 後に新規 ST を発行
- ST が Timeout している時 or ST が Valid でない時
 - Login への redirection を発行
 - Authorization 後に新規 ST を発行
- “Login への redirection” = “TGC の検証” + “Authorization”

CAS 認証のしくみ (2: Verify Ticket (0))

LDAP Server

Web Application

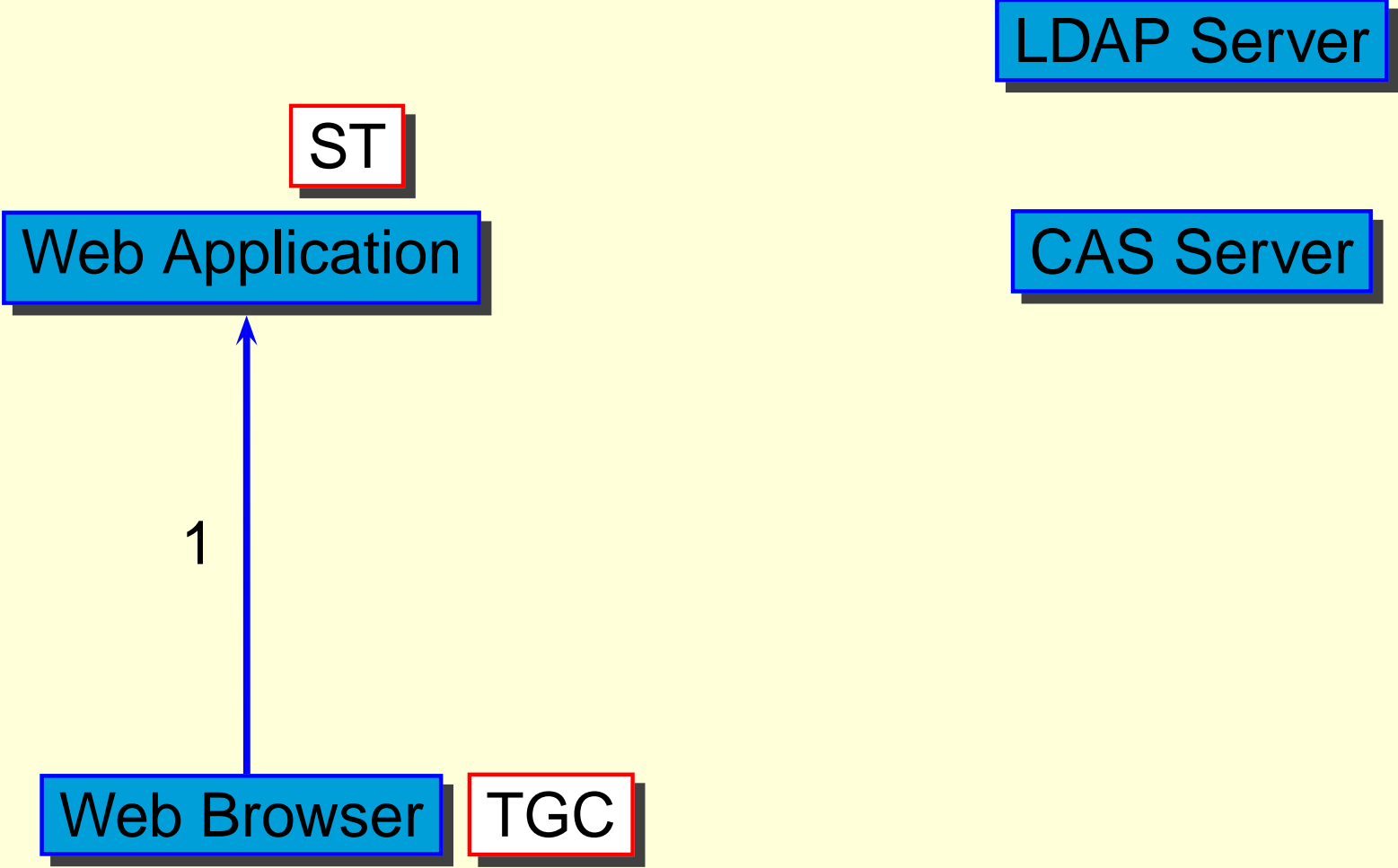
CAS Server

Web Browser

TGC

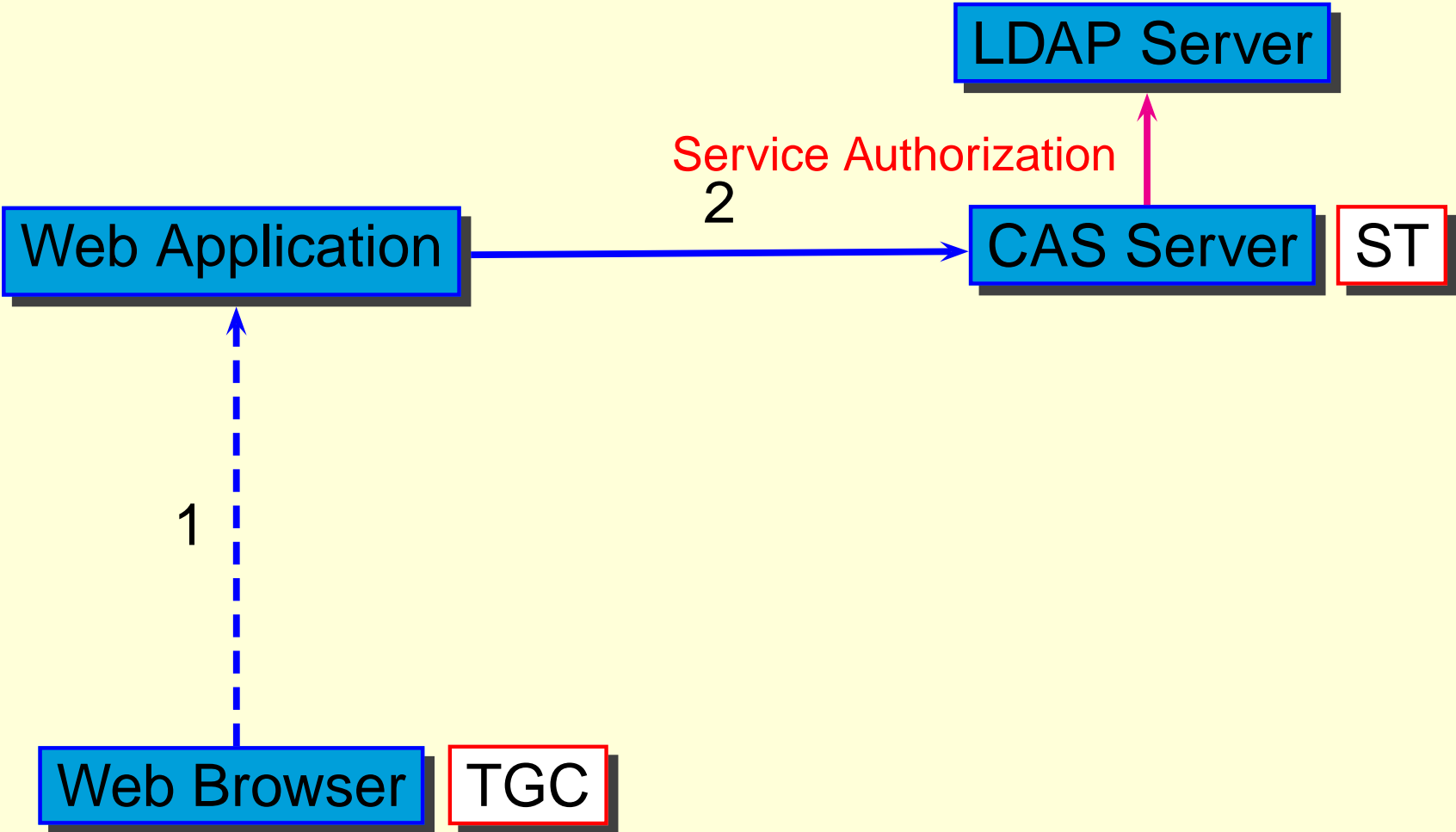
ST

CAS 認証のしくみ (2: Verify Ticket (1))



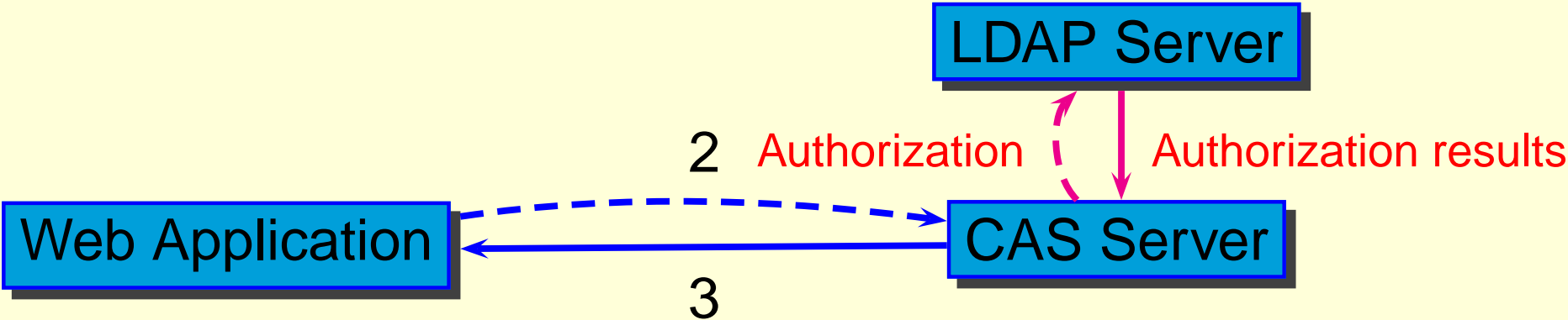
1. Access to <https://aFQDN/a.html&ticket=ST-xxxxx>

CAS 認証のしくみ (2: Verify Ticket (2))



2. Verify `ticket=ST-xxxxx` with `service=https://aFQDN/a.html`

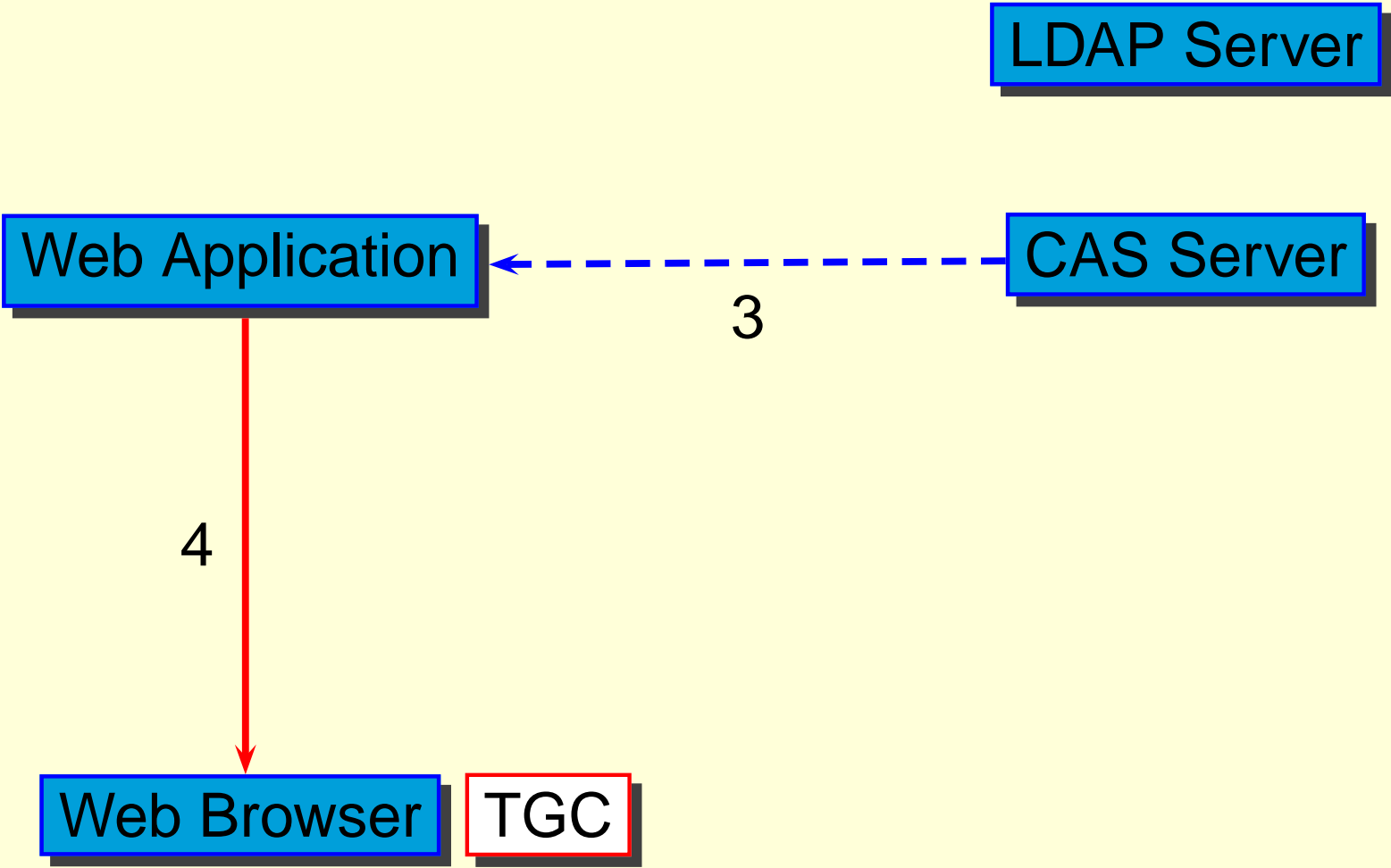
CAS 認証のしくみ (2: Verify Ticket (3))



Web Browser TGC

3. Get authorization results and user information

CAS 認証のしくみ (2: Verify Ticket (4))



4. Reply from Web Application

Verify Ticket の注意事項

- Verify Ticket の結果は CAS client が処理する
- CAS client が返すもの (Original の CAS)
 - Ticket Validation の結果
 - ユーザ ID
- CAS client が返すもの (CAS²)
 - Ticket Validation の結果
 - CAS-ACL で指定されたユーザデータベース内の属性値
 - 各 Web Application は CAS client module からの戻り値を見ることでユーザの属性値を取得

CAS 認証のしくみ (3: Access to another Application)

- Ticket Granting Ticket を持っている状態で「他のアプリケーション」にアクセスする
 - より正確には、「他の Access Control Class の URL」にアクセスする
 - 有効な Service Ticket が存在しない
 - Service Ticket が Timeout している
- Ticket Granting Cookie を検証
- 新規 Service Ticket の発行

CAS 認証のしくみ (3: Access to another Application (0))

LDAP Server

Web Application

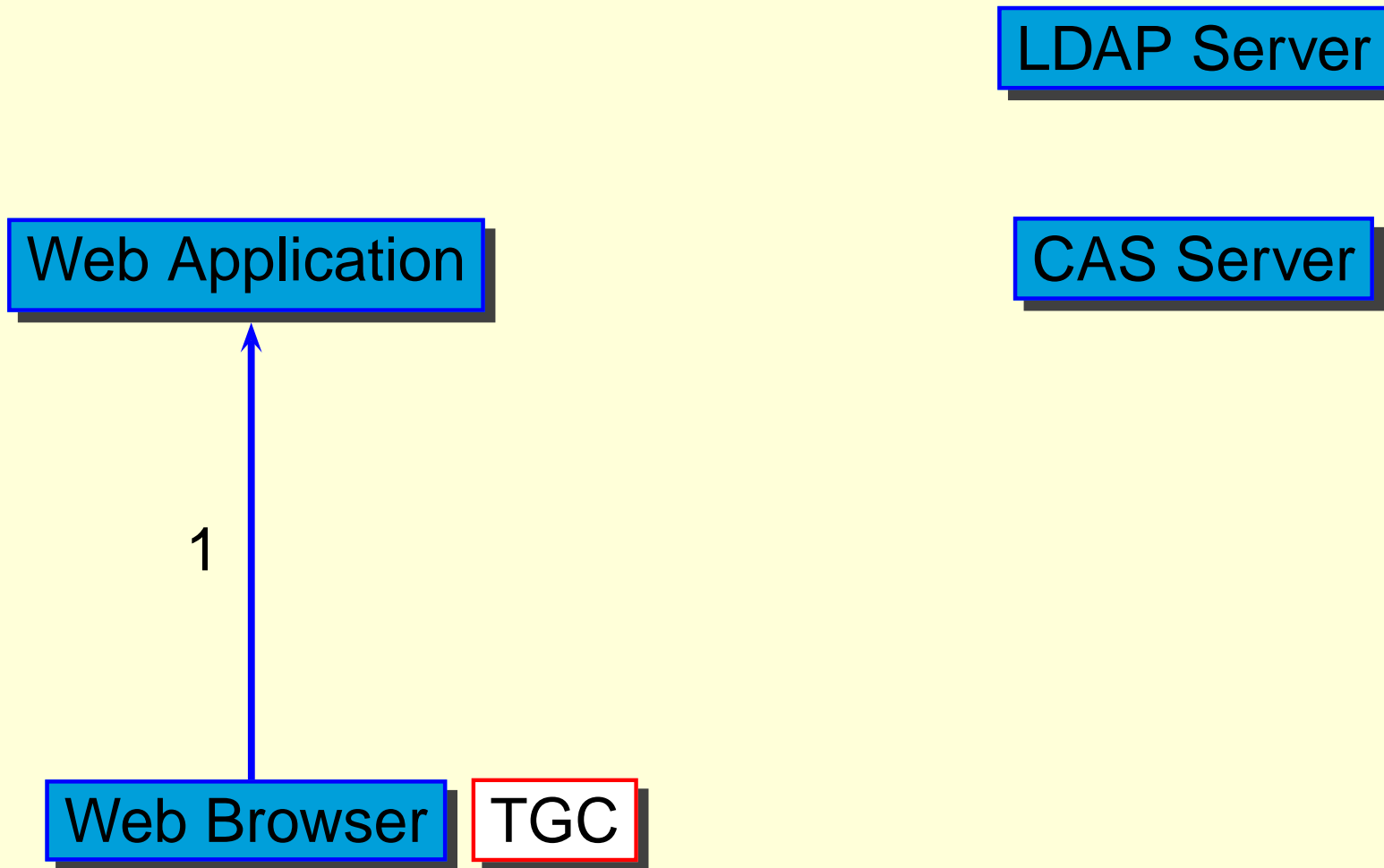
CAS Server

Web Browser

TGC

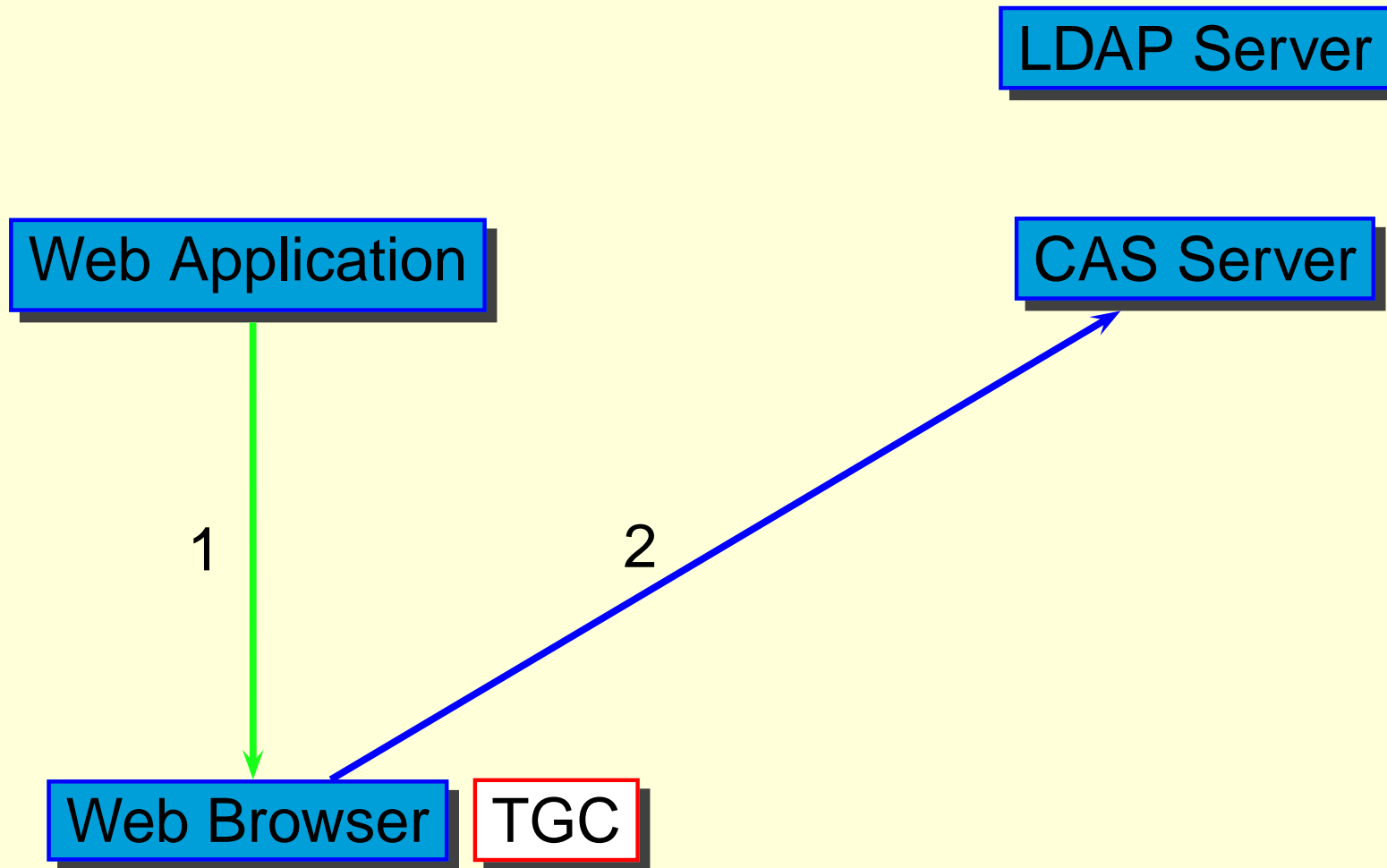
no ST, ST is expired or ST is belonged to different ACCESS CLASS

CAS 認証のしくみ (3: Access to another Application (1))



1. Access to <https://aFQDN/a.html>

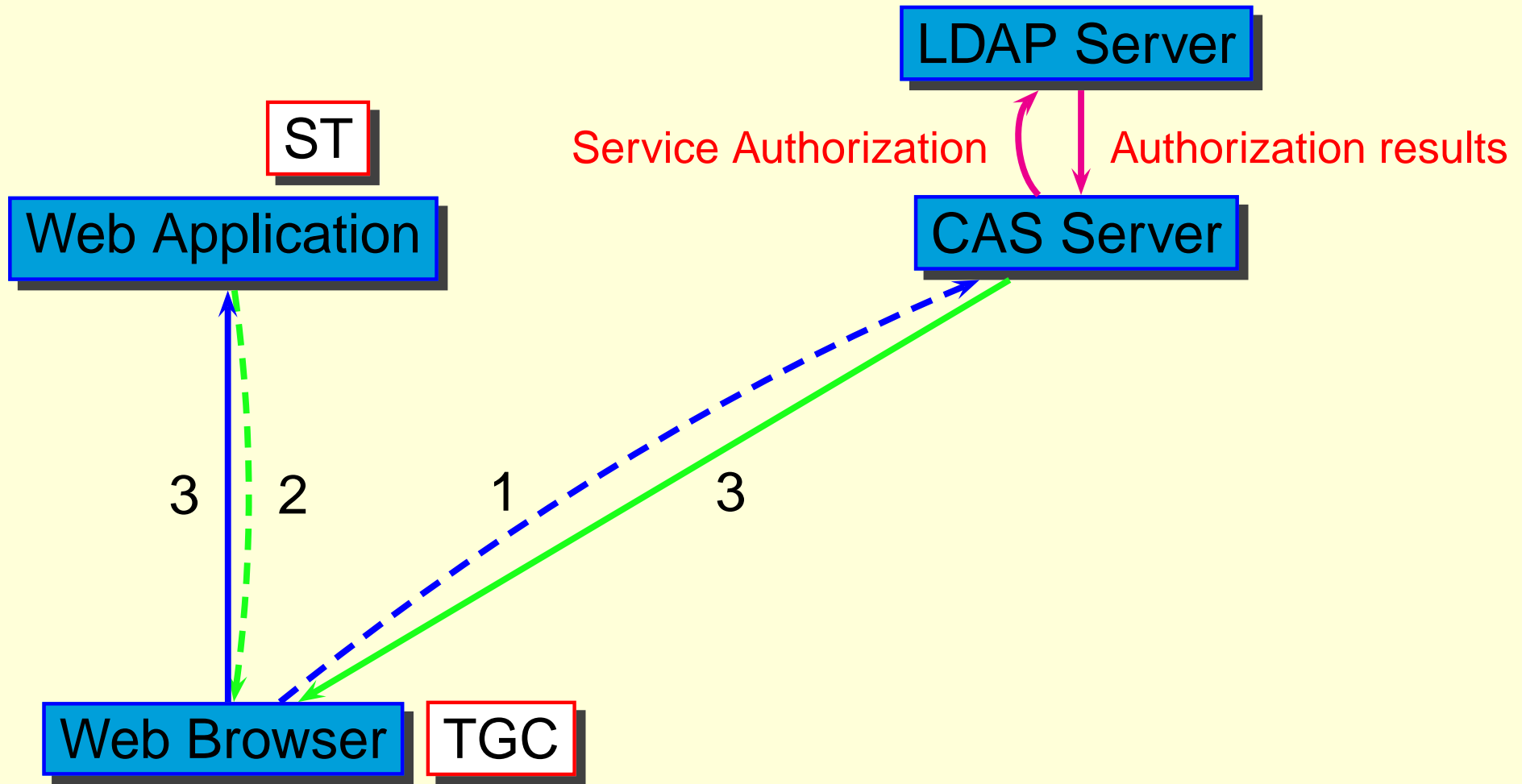
CAS 認証のしくみ (3: Access to another Application (2))



4. Redirect to

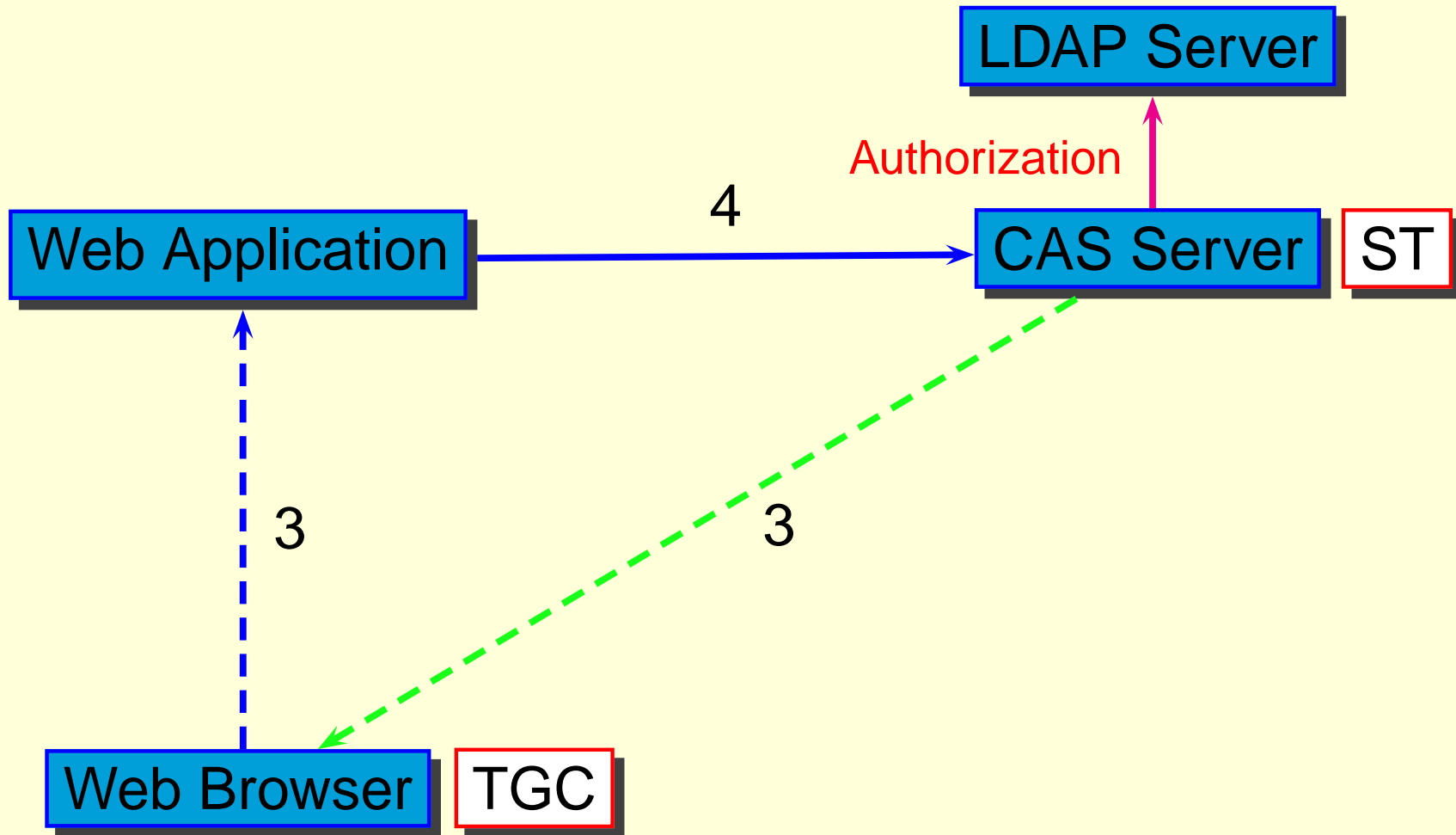
<https://CAS/login&service=https://aFQDN/a.html>

CAS 認証のしくみ (3: Access to another Application (3))



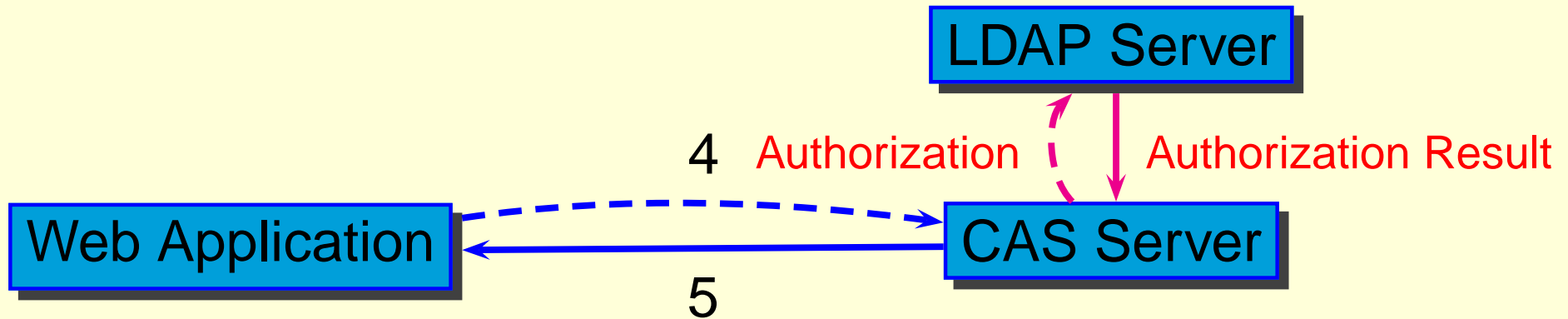
5. Redirect to <https://aFQDN/a.html&ticket=ST-xxx>

CAS 認証のしくみ (3: Access to another Application (4))



6. Verify Service Ticket

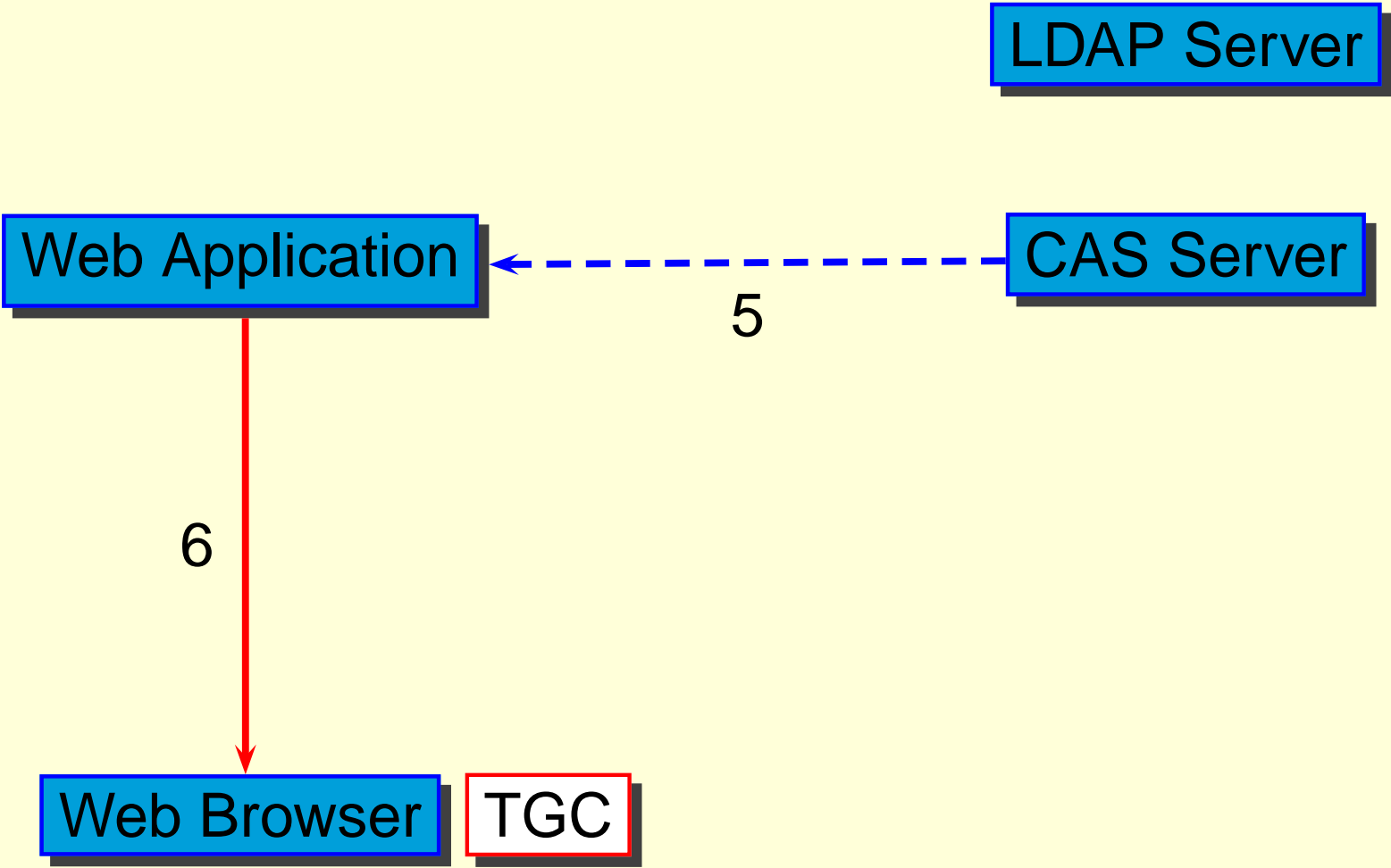
CAS 認証のしくみ (3: Access to another Application (5))



Web Browser TGC

7. Receive verify result form CAS server

CAS 認証のしくみ (3: Access to another Application (6))



8. Receive Data from Application Server

CAS2 のその他の機能

- CAS-ACL の分散管理
 - CAS-ACL の subTree ごとに管理者を設定可能
- Alart 機能
 - Login 時にユーザに応じた情報を提示可能
 - 名古屋大学ポータルでは「電子メールアドレスの登録の要請」を実施
- 簡便な「セッション維持」機能
 - 同一 CAS-ACC に属するアクセスに対しては「次のアクセスのための ST」(nextticket) を発行可能
- 二重ログインの検出
- 「Mail Address」(など) を User ID の代りに利用可能
- Post メソッドへの対応

CAS の利点

- Single Sign On 環境を容易に実現できる
- Web Application 側には CAS client module を追加するだけ
- Web Application はユーザの認証情報を受け取らない
- Web Application が認証データベースへ直接アクセスしない
- 暗号化は SSL のみを利用
- 軽くて高速に動作
 - 最大アクセス実績：4000 回/分
 - Sun Fire V480 (1.0GHz UltraSPAC III Cu x 2)
 - 4.0GB Memory
 - Solaris 8

将来にむけて

- クライアント証明書などへの対応
- 非 Web Application での利用方法の開発
- Federated CAS の開発
- CAS Version 3 への対応
- 現在の状況
 - Open Source での公開に向けて準備中
 - CAS-Client などのクラスライブラリのドキュメントなどを作成中
 - CAS-ACL 管理 Application の開発

参考文献・URL

- <http://www.math.nagoya-u.ac.jp/~naito/cas/>
CAS² の各種情報
- <http://www.math.nagoya-u.ac.jp/~naito/cas/javadoc/>
JavaDoc for CAS² 関連の Java class library
- 文献
 - 内藤-梶田, 京都大学数理解析研究所講究録, Vol. 1446, 14–39, (2005)
 - 内藤-梶田-小尻-平野-間瀬, 情報処理学会論文誌, Vol. 47-4 (掲載予定)

● お試し CAS²

- <http://tomcat.math.nagoya-u.ac.jp/casfreetest/>
(Java Servlet)
- http://tomcat.math.nagoya-u.ac.jp/casfreetest_jsp/
(JSP)
- <http://www.math.nagoya-u.ac.jp/~naito/casfreetest/>
(Perl SSI)

● 3つの Web Application が Single Sign On で利用可能.

- ユーザ ID : cas0, cas1, ..., cas9
パスワード : 「ユーザ ID と同一の文字列」
- Perl SSI ページだけは cas9 はアクセス不可
- Perl SSI ページに cas9 アクセスすると「アクセスできません」と表示される

Graduate School of Mathematics
Nagoya University Central Authentication System

User ID :

Password:

他のログインセッションを無効にする
 登録したメールアドレスを変更する

Login

- “Mail Address” でもログイン可能. (`cas0@math.nagoya-u.ac.jp` など. パスワードは同じ)
- 「他のログインセッションを無効にする」とは, 他のブラウザから同一ID でログインしているとき, 他のブラウザからのセッションを無効にして, このブラウザからログインすること. (今回は「登録したメールアドレスを変更する」は機能しない)
- Source Code, CAS-ACL は

<http://www.math.nagoya-u.ac.jp/~naito/casfreetest/source/> 参照