Załącznik nr 1d do SIWZ

Protokół CAS

Autor: Drew Mazurek
Pomoc: Susan Bramhall, Howard Gilbert, Andy Newman, Andrew Petro

Wersja 1.0
Data publikacji: 4 maja 2005 r.

1. Wstęp

Jest to oficjalna specyfikacja protokołów CAS 1.0 i 2.0. Mogą być do niej wprowadzane zmiany.

1.1. Terminy i definicje

Użyte w tym dokumencie słowa kluczowe: „MUSI", „NIE MOŻE", „WYMAGANY", „POWINIEN", „NIE POWINIEN", „MOŻE" oraz „OPCJONALNY" należy interpretować zgodnie z wykładnią RFC 2119[1].

„Klient" oznacza użytkownika końcowego i/lub przeglądarkę internetową
„Serwer" oznacza serwer *Centralnych usług uwierzytelniania*
„Usługa" oznacza aplikację, do której dostęp chce uzyskać klient
„Usługa wewnętrzna" oznacza aplikację, do której usługa chce uzyskać dostęp w imieniu klienta. Określana również jako „usługa docelowa".
<LF> oznacza koniec linii tekstu (wartość ASCII: 0x0a)

2. Adres URI (jednolity identyfikator zasobu) w systemie CAS

CAS to protokół oparty na HTTP[2,3], wymagający, by każdy z jego składników dostępny był z poziomu określonych URI. W tym dziale omówiono każdy URI.

2.1. Funkcja /login jako żądający danych identyfikacyjnych

URI funkcji /login ma podwójny charakter: pozwala logować się jako żądający danych identyfikacyjnych i jako przyjmujący dane identyfikacyjne. W odpowiedzi na dane identyfikacyjne pracuje w trybie przyjmującego te dane. W przeciwnym wypadku działa jako żądający danych identyfikacyjnych.

Jeśli klient zalogował się już za pomocą CAS do jednej sesji, przeglądarka internetowa przesyła do CAS zabezpieczony plik cookie z ciągiem znaków identyfikującym tzw. „bilet na bilet" (ticket-granting ticket, TGT). Jest on nazywany plikiem cookie udostępniającym bilet. Jeśli plik cookie udostępniający bilet pasować będzie do poprawnego biletu na bilet, system CAS wyda bilet na usługę pod warunkiem, że spełnione zostaną wszystkie pozostałe warunki wymienione w tej specyfikacji. Więcej informacji o plikach cookie udostępniających bilet znaleźć można w dziale 3.6.

2.1.1. Parametry

Niżej wymienione parametry żądania HTTP mogą zostać podane w polu logowania, gdy loguje się żądający uwierzytelnienia. Przy ich wprowadzaniu istotna jest wielkość liter i MUSZĄ one być zarządzane przez /login.

- service [OPCJONALNY] – identyfikator aplikacji, do której dostęp stara się uzyskać klient. Prawie zawsze będzie to adres URL aplikacji. Proszę zauważyć, że będąc parametrem żądania HTPP, wprowadzony adres URL MUSI być zaszyfrowany jako URL, zgodnie z opisem w dziale 2.2 RFC 1738[4]. Jeśli nie wskazano usługi i nie uruchomiono sesji pojedynczego logowania, system CAS powinien poprosić użytkownika o podanie danych uwierzytelniających, aby uruchomić taką sesję. Jeśli nie wskazano usługi, ale uruchomiono sesję pojedynczego logowania, system CAS powinien wyświetlić komunikat z informacją, że klient jest już zalogowany.
- renew [OPCJONALNY] – po ustawieniu tego parametru pojedyncza sesja logowania zostanie pominięta. W takim przypadku system CAS zażąda od klienta podania danych uwierzytelniających bez względu na to, czy zainicjowano już sesję pojedynczego logowania w ramach CAS. Parametr ten nie jest kompatybilny z parametrem „gateway" („brama"). Usługi przekierowujące do URI funkcję /login oraz przesyłające do niej dane z formularza nie powinny wprowadzać parametrów żądania „renew" ani „gateway". Jeśli wprowadzone zostaną oba parametry, tryb pracy nie będzie zdefiniowany. Zaleca się, by implementacje CAS ignorowały parametr „gateway", jeśli wprowadzono parametr „renew". Zaleca się także, by w przypadku wprowadzenia parametru „renew" ustawić jego wartość na „true" („prawda").

- gateway („brama") [OPCJONALNY] – po wprowadzeniu tego parametru system CAS nie będzie żądał od klienta podania danych uwierzytelniających. Jeśli klient już wcześniej uruchomił sesję pojedynczego logowania w systemie CAS lub jeśli taką sesję można uruchomić metodami nieinteraktywnymi (w oparciu o procedurę uwierzytelnienia zaufania), system CAS może przekierować klienta do adresu URL wskazanego przez parametr „usługa", załączając ważny bilet na usługę (system może też wyświetlić stronę informującą klienta, że przeprowadzono procedurę uwierzytelnienia). Jeśli klient nie uruchomił sesji pojedynczego logowania w systemie CAS ani nie da się zainicjować autentykacji nieinteraktywnej, system musi przekierować klienta na adres URL wskazany przez parametr „usługa", nie załączając parametru „bilet" to adresu URL. Jeśli nie wprowadzono parametru „usługa", a wprowadzono parametr „gateway", działanie systemu CAS będzie niezdefiniowane. Zaleca się, by w takim przypadku system zażądał podania danych uwierzytelniających, tak jakby nie wprowadzono żadnego parametru. Parametr ten nie jest kompatybilny z parametrem „renew". Wprowadzenie obu tych parametrów skutkuje niezdefiniowanym działaniem systemu. Zaleca się ponadto, by przy wprowadzaniu parametru „gateway" ustawić jego wartość na „true" („prawda").

2.1.2. Przykładowe adresy URL dla funkcji /login

Przykładowy login:

```
https://server/cas/login?service=http%3A%2F%2Fwww.service.com
```

Bez żądania nazwy użytkownika / hasła:

```
https://server/cas/login?service=http%3A%2F%2Fwww.service.com&renew=true
```

Zawsze z żądaniem nazwy użytkownika / hasła:

```
https://server/cas/login?service=http%3A%2F%2Fwww.service.com&renew=true
```

2.1.3. Odpowiedź przy uwierzytelnianiu nazwy użytkownika/hasła

Jeśli loguje się żądający uwierzytelnienia, odpowiedź będzie się różnić w zależności od rodzaju żądanych danych uwierzytelniających. W większości przypadków CAS odpowie wyświetlając ekran logowania z polami na nazwę użytkownika i hasło. Strona ta musi zawierać formularz z parametrami „username" („użytkownik"), „password" („hasło") i „lt". W formularzu może też znajdować się parametr „warn" („uwaga). Jeśli parametr „service" przypisano do funkcji /login, musi on być także parametrem formularza i

zawierać wartość pierwotnie wprowadzoną do tego pola. Parametry te omówiono szczegółowo w dziale 2.2.1. Formularz musi zostać przesłany za pomocą metody POST HTTP do funkcji /login, który będzie wówczas działał jako przyjmujący dane uwierzytelniające (więcej informacji w dziale 2.2).

2.1.4. Odpowiedź na uwierzytelnienie zaufania

Procedura uwierzytelnienia zaufania uwzględnia arbitralne aspekty żądania jako podstawę uwierzytelnienia. Doświadczenia użytkownika związane z procedurą uwierzytelniania zaufania będą ściśle zależne od danego przypadku, w tym lokalnych przepisów i systemu logistycznego dot. zastosowanego mechanizmu uwierzytelniania. Jeśli funkcja /login przy uwierzytelnianiu zaufania działa jak żądający danych identyfikacyjnych, jego działanie zależeć będzie od rodzaju danych identyfikacyjnych, które otrzyma. Jeśli dane te będą poprawne, system CAS może w sposób transparentny przekierować użytkownika do usługi. Opcjonalnie system może wyświetlić komunikat informujący, że dane identyfikacyjne zostały wprowadzone i zezwolić klientowi na potwierdzenie, że chce tych danych użyć. Zaleca się, by system CAS pozwolił użytkownikowi wybrać preferowany tryb pracy. Jeśli dane identyfikacyjne są błędne lub nie istnieją, zaleca się, by system CAS wyświetlił komunikat z informacją o przyczynach nieudanego uwierzytelnienia i ewentualnie wskazał inną metodę (uwierzytelnienie np. w oparciu o nazwę użytkownika / hasło).

2.1.5. Odpowiedź w przypadku pojedynczego uwierzytelnienia

Jeśli klient zainicjował już sesję pojedynczego uwierzytelnienia w systemie CAS, jego plik cookie z sesji HTTP zostanie przekazany do funkcji /login, a tryb pracy będzie zarządzany zgodnie z opisem w dziale 2.2.4. Jeśli jednak ustawiono parametr „renew", tryb pracy będzie zarządzany zgodnie z opisem w rozdziale 2.1.3 lub 2.1.4.

2.2. Funkcja /login jako przyjmujący dane identyfikacyjne

W przypadku przesłania do funkcji /login zbioru danych identyfikacyjnych /login działa w trybie przyjmującego dane identyfikacyjne. Ten tryb pracy opisano w bieżącym rozdziale.

2.2.1. Parametry wspólne dla wszystkich rodzajów uwierzytelnienia

Poniższe parametry żądania HTTP mogą zostać przesłane do funkcji /login, gdy działa on w trybie przyjmującego dane identyfikacyjne. Przy ich wprowadzaniu istotna jest wielkość liter i każdy z nich musi być zarządzany przez /login.

- service („usługa") [OPCJONALNY] – adres URL aplikacji, do której klient chce uzyskać dostęp. Po udanym uwierzytelnieniu CAS musi przekierować klienta na ten adres. Więcej informacji na ten temat znaleźć można w rozdziale 2.2.4.
- warn („ostrzeżenie") – jeśli ustawiono ten parametr, pojedyncza sesja logowania nie może być transparentna. Klient musi otrzymać powiadomienie przed uzyskaniem dostępu do innej usługi.

2.2.2. Parametry uwierzytelniania opartego na nazwie użytkownika / haśle

Oprócz opcjonalnych parametrów wymienionych w rozdziale 2.2.1, następujące parametry żądania HTTP muszą zostać przesłane do funkcji /login, gdy działa on w trybie przyjmującego dane identyfikacyjne przy uwierzytelnianiu opartym na nazwie użytkownika/haśle. Przy ich wprowadzaniu istotna jest wielkość liter.

- username („nazwa użytkownika") [WYMAGANY] – nazwa użytkownika, którą podaje klient chcący się zalogować
- password („hasło") [WYMAGANY] – hasło wprowadzane przez klienta, który chce się zalogować
- lt [WYMAGANY] – bilet logowania. Część formularza logowania omówionego w rozdziale 2.1.3. Sam bilet logowania opisano w rozdziale 3.5.

2.2.3. Parametry uwierzytelniania zaufania

Żadne parametry żądania HTTP nie są wymagane do uwierzytelniania zaufania. Procedura ta może się odbyć w oparciu o dowolny aspekt żądania HTTP.

2.2.4. Odpowiedź

Jeśli /login działa w trybie przyjmującego dane identyfikacyjne, musi mu zostać przesłana jedna z następujących odpowiedzi:

- Udane logowanie: przekierowanie klienta na adres URL podany w parametrze „service" („usługa") tak, aby dane identyfikacyjne klienta nie zostały przesłane do usługi. Wynikiem przekierowania musi być wysłanie przez klienta żądania GET do usługi. Żądaniu musi towarzyszyć poprawny bilet usługi, przesłany jako parametr żądania HTTP „ticket" („bilet"). Więcej informacji na ten temat znaleźć można w załączniku B. Jeśli parametru „service" nie podano, system CAS musi wyświetlić komunikat informujący klienta, że udało się uruchomić sesję pojedynczego logowania.
- Nieudane logowanie: powrót do funkcji /login jako żądający danych identyfikacyjnych. Zaleca się w takim przypadku, by serwer CAS wyświetlił komunikat informujący o błędzie, z podaniem przyczyn nieudanego logowania (błędne hasło, zablokowane konto itp.) oraz, w stosownych sytuacjach, umożliwił użytkownikowi podjęcie ponownej próby logowania.

2.3. Funkcja /logout

Funkcja /logout kończy sesję pojedynczego logowania w systemie CAS. Plik cookie udostępniający bilet (rozdział 3.6) zostaje skasowany, a kolejne żądania wysyłane do funkcji /login będą wymagać od użytkownika ponownego wprowadzenia podstawowych danych identyfikacyjnych, zanim uzyska on bilety usługi (tym samym zostanie ustanowiona nowa sesja pojedynczego logowania).

2.3.1. Parametry

Do opcji /logout może zostać przypisany następujący parametr żądania HTTP. Przy jego wprowadzaniu istotna jest wielkość liter i musi on być zarządzany przez /logout.

- url [OPCJONALNY] – jeśli wprowadzono parametr „url", adres URL z nim skojarzony powinien widnieć na stronie wylogowania wraz z opisem. Na przykład: „Aplikacja, z której się właśnie wylogowałeś, prosi Cię o kliknięcie wysłanego przez nią odnośnika. Kliknij tutaj, aby wejść na stronę http://www.go-back.edu".

2.3.2. Odpowiedź

Funkcja /logout musi wyświetlić stronę z informacją o tym, że użytkownik został wylogowany. Jeśli wprowadzono parametr żądania „url", /logout powinien także podać odnośnik do adresu URL, zgodnie z opisem w rozdziale 2.3.1.

2.4. Funkcja /validate [CAS 1.0]

Funkcja /validate weryfikuje poprawność biletu usługi. Opcja ta należy do protokołu CAS 1.0, w związku z czym nie obsługuje uwierzytelniania pośredniego. Po przesłaniu do /validate biletu pośredniego system CAS musi odpowiedzieć informacją o nieudanej weryfikacji biletu.

2.4.1. Parametry

Do funkcji /validate muszą zostać przypisane następujące parametry. Przy ich wprowadzaniu istotna jest wielkość liter i muszą one być zarządzane przez /validate.

Service („usługa") [WYMAGANY] – identyfikator usługi, do której wydano bilet, zgodnie z opisem w rozdziale 2.2.1.

Ticket („bilet") [WYMAGANY] – bilet usługi wydany przez /login. Biletom usług poświęcony jest rozdział 3.1.

Renew („ponów") [OPCJONALNY] – jeśli wprowadzono ten parametr, weryfikacja biletu powiedzie się wyłącznie, jeśli bilet usługi został wydany po przedstawieniu przez użytkownika podstawowych danych identyfikacyjnych (a nie w ramach sesji pojedynczego logowania).

2.4.2. Odpowiedź

Funkcja /validate wyśle jedną z dwóch odpowiedzi zwrotnych:
Po udanej weryfikacji biletu:

```
yes<LF>
username<LF>
```

Po nieudanej weryfikacji biletu:

```
no<LF>
<LF>
```

2.4.3. Przykładowe adresy URL do funkcji /validate

Prosta próba weryfikacji:

```
https://server/cas/validate?service=http%3A%2F%2Fwww.service.com&ticket=.
...
```

Upewnienie się, że bilet usługi został wydany po przedstawieniu podstawowych danych identyfikacyjnych:

```
https://server/cas/validate?service=http%3A%2F%2Fwww.service.com&ticket=.
...
```

2.5. /serviceValidate [CAS 2.0]

Funkcja /serviceValidate poddaje weryfikacji bilet usługi i wysyła odpowiedź zwrotną w postaci fragmentu kodu XML.
Na żądanie /serviceValidate musi też wygenerować i wydać tzw. proxy-granting tickets („bilety nadające pośrednictwo"). /serviceValidate nie może uznać weryfikacji za

udaną, jeśli otrzyma bilet pośredni. Zaleca się, by w przypadku otrzymania przez /serviceValidate biletu usługi, komunikat o błędzie w odpowiedzi XML zawierał informację o tym, że weryfikacja nie powiodła się, ponieważ do /serviceValidate został przesłany bilet.

2.5.1. Parametry

Do funkcji /serviceValidate mogą być przypisane następujące parametry żądania HTTP. Przy ich wprowadzaniu istotna jest wielkość liter i muszą one być zarządzane przez /serviceValidate.

- service [WYMAGANY] – identyfikator usługi, dla której wydano bilet, zgodnie z opisem w rozdziale 2.2.1.
- ticket [WYMAGANY] – bilet usługi wydany przez /login. Biletom usług poświęcono rozdział 3.1.
- pgtUrl [OPCJONALNY] – adres URL pośredniego wywołania zwrotnego. Omówione w rozdziale 2.5.4.
- renew [OPCJONALNY] – jeśli wprowadzono ten parametr, weryfikacja biletu powiedzie się wyłącznie, jeśli bilet usługi został wydany po przedstawieniu przez użytkownika podstawowych danych identyfikacyjnych (a nie w ramach sesji pojedynczego logowania).

2.5.2. Odpowiedź

Funkcja /serviceValidate zwróci odpowiedź usługi CAS („CAS serviceResponse") w formacie XML, zgodnie z opisem struktury XML w załączniku A. Poniżej przedstawiono przykładowe odpowiedzi:
Po udanej weryfikacji biletu:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
    <cas:authenticationSuccess>
        <cas:user>username</cas:user>
            <cas:proxyGrantingTicket>PGTIOU-84678-8a9d...
        </cas:proxyGrantingTicket>
    </cas:authenticationSuccess>
</cas:serviceResponse>
```

Po nieudanej weryfikacji biletu:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
    <cas:authenticationFailure code="INVALID_TICKET">
        Ticket ST-1856339-aA5Yuvrxzpv8TaulcYQ7 not recognized
    </cas:authenticationFailure>
</cas:serviceResponse>
```

2.5.3. Kody błędów

Poniższe wartości mogą zostać użyte jako atrybuty „kodowe" odpowiedzi informujących o nieudanej weryfikacji. Jest to minimalna lista kodów błędów, które muszą obsługiwać wszystkie serwery CAS. Poszczególne implementacje mogą obsługiwać też kody innego rodzaju.

- INVALID_REQUEST – nie wszystkie wymagane parametry żądania są dostępne.
- INVALID_TICKET – wprowadzony bilet nie jest poprawny lub nie pochodzi z pierwszego logowania, a przy weryfikacji ustawiono parametr „renew". W odpowiedzi XML, w sekcji zaczynającej się od <cas:authenticationFailure> powinny znajdować się szczegółowe informacje na ten temat.
- INVALID_SERVICE – wprowadzono poprawny bilet, ale wskazana usługa była niezgodna z usługą przypisaną do biletu. System CAS musi unieważnić bilet i wykluczyć go z kolejnych weryfikacji.
- INTERNAL_ERROR – podczas weryfikacji biletu wystąpił błąd wewnętrzny.

W przypadku wszystkich kodów błędów zaleca się, by w głównej części sekcji <cas:authenticationFailure> w odpowiedzi XML system CAS zamieścił szczegółowe informacje na ich temat.

2.5.4. Wywołanie zwrotne pośrednie

Jeśli usługa chce zlecić pośredniczenie w uwierzytelnieniu klienta usłudze wewnętrznej, musi uzyskać bilet nadający pośrednictwo (proxy-granting ticket). Procedura pozyskiwania tego biletu zarządzana jest w oparciu o pośredni adres URL wywołania zwrotnego. Adres ten pozwoli jednoznacznie i bezpiecznie zidentyfikować usługę wewnętrzną, która pośredniczy w uwierzytelnieniu klienta. Usługa wewnętrzna może wówczas zdecydować, czy akceptuje dane identyfikacyjne na podstawie jej identyfikującego URL wywołania zwrotnego.
Mechanizm pośredniczącego wywołania zwrotnego działa w następujący sposób:

- Usługa, która żąda wydania biletu nadającego pośrednictwo, przy pierwszym bilecie usługi lub weryfikacji biletu pośredniego ustawia parametr żądania HTTP „pgtUrl" w opcji /serviceValidate (lub /proxyValidate). Jest to URL wywołania

zwrotnego dla usługi, z którą CAS się połączy w celu weryfikacji tożsamości usługi. Musi to być URL typu HTTPS, a system CAS musi sprawdzić zarówno czy certyfikat SSL jest poprawny, jak również czy jego nazwa zgodna jest z nazwą usługi. Jeśli certyfikat nie przejdzie weryfikacji, nie zostanie wydany bilet nadający pośrednictwo i w takim przypadku odpowiedź usługi CAS, zgodnie z opisem w rozdziale 2.5.2., nie może zawierać sekcji <proxyGrantingTicket>. Na tym etapie procedura wydania biletu nadania pośrednictwa zostanie wstrzymana, ale weryfikacja biletu usługi będzie kontynuowana i użytkownik otrzyma informację o jej pozytywnym lub negatywnym wyniku. Jeśli certyfikat zostanie zweryfikowany pozytywnie, procedura wydania biletu nadającego pośrednictwo będzie kontynuowana w sposób opisany w kroku nr 2.

- System CAS stosuje żądanie HTTP GET w celu przesłania parametrów żądania HTTP „pgtId" i „pgtIou" do pgtUrl. Elementy te opisano odpowiednio w rozdziałach 3.3 i 3.4.
- Jeśli HTTP GET zwróci kod statusu HTTP 200 (OK), system CAS musi zareagować na żądanie /serviceValidate (lub /proxyValidate) odpowiedzią usługi (rozdział 2.5.2) wraz z IOU biletu nadającego pośrednictwo (rozdział 3.4) w sekcji <cas:proxyGrantingTicket>. Jeśli żądanie HTTP GET zwróci inny kod statusu, z wyjątkiem przekierowań HTTP 3xx, system CAS musi zareagować na żądanie /serviceValidate (lub /proxyValidate) odpowiedzią usługi, która nie może zawierać sekcji <cas:proxyGrantingTicket> System CAS może prześledzić wszelkie przekierowania HTTP pochodzące z parametru pgtUrl. Identyfikujący adres URL wywołania zwrotnego umieszczany przy weryfikacji w sekcji <proxy> musi jednak być tym samym adresem URL, który na początku przesłany został do /serviceValidate (lub /proxyValidate) jako parametr „pgtUrl".
- Po otrzymaniu IOU biletu nadającego pośrednictwo w odpowiedzi CAS, jak również biletu nadającego pośrednictwo i IOU biletu nadającego pośrednictwo z wywołania zwrotnego, usługa skorzysta z IOU biletu nadającego pośrednictwo do skorelowania tego biletu z odpowiedzią weryfikacyjną. Następnie usługa użyje biletu nadającego pośrednictwo do pozyskania biletów pośrednich, zgodnie z opisem w rozdziale 2.7.

2.5.5. Przykładowe adresy URL w /serviceValidate

Prosta próba weryfikacji:

```
https://server/cas/serviceValidate?service=http%3A%2F%2Fwww.service.com&.
...
```

Upewnienie się, że bilet usługi został wydany po przedstawieniu podstawowych danych identyfikacyjnych:

```
https://server/cas/serviceValidate?service=http%3A%2F%2Fwww.service.com&.
.. ST-1856339-aA5Yuvrxzpv8Tau1cYQ7&renew=true
```

Wprowadzenie URL wywołania zwrotnego na potrzeby pośredniczenia:

```
https://server/cas/serviceValidate?service=http%3A%2F%2Fwww.service.com&.
..
```

2.6. /proxyValidate [CAS 2.0]

Funkcja /proxyValidate musi realizować te same zadania związane z weryfikacją, jak /serviceValidate, a ponadto prowadzić weryfikację biletów pośrednich. Funkcja /proxyValidate musi być w stanie weryfikować zarówno bilety usług, jak i bilety pośrednie.

2.6.1. Parametry

Funkcja /proxyValidate ma identyczne wymogi parametrowe, jak funkcja /serviceValidate. Patrz rozdział 2.5.1.

2.6.2. Odpowiedź

Funkcja /proxyValidate zwróci odpowiedź usługi CAS („CAS serviceResponse") w formacie XML, zgodnie z opisem struktury XML w załączniku A. Poniżej przedstawiono przykładowe odpowiedzi:

Po udanej weryfikacji biletu:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
    <cas:authenticationSuccess>
        <cas:user>username</cas:user>
        <cas:proxyGrantingTicket>PGTIOU-84678-8a9d...</cas:proxyGrantingT
icket>
        <cas:proxies>
            <cas:proxy>https://proxy2/pgtUrl</cas:proxy>
            <cas:proxy>https://proxy1/pgtUrl</cas:proxy>
        </cas:proxies>
    </cas:authenticationSuccess>
</cas:serviceResponse>
```

Uwaga: jeśli weryfikacja odbywała się za pośrednictwem wielu pośredników (proxies), ich kolejność powinna być zachowana w sekcji <cas:proxies>. Jako pierwszy na liście musi widnieć ostatni aktywny pośrednik, a w miarę dodawania nowych pośredników lista musi przesuwać się w dół. W przykładzie powyżej pośrednik najpierw obsłużył usługę oznaczoną jako https://proxy1/pgtUrl i usługa ta zleciła pośredniczenie w weryfikacji usłudze oznaczonej jako https://proxy2/pgtUrl.

Po nieudanej weryfikacji biletu:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
    <cas:authenticationFailure code="INVALID_TICKET">
        ticket PT-1856376-1HMgO86Z2ZKeByc5XdYD not recognized
    </cas:authenticationFailure>
</cas:serviceResponse>
```

2.6.3. Przykładowe adresy URL funkcji /proxyValidate

Funkcja /proxyValidate obsługuje te same parametry, co /serviceValidate. Rozdział 2.5.5 zawiera przykłady użycia tej funkcji („serviceValidate" należy zamienić na „proxyValidate").

2.7. Funkcja /proxy [CAS 2.0]

Funkcja /proxy udostępnia bilety pośrednie usługom, które otrzymały bilety nadania pośrednictwa i będą zlecać pośrednictwo uwierzytelniania usługom wewnętrznym.

2.7.1. Parametry

Poniższe parametry żądania HTTP muszą zostać ustanowione dla funkcji /proxy. Przy ich wprowadzaniu istotna jest wielkość liter.

- pgt [WYMAGANY] – bilet nadania pośrednictwa uzyskany przez usługę w trakcie weryfikacji biletu usługi lub biletu pośredniego
- targetService [WYMAGANY] – identyfikator usługi wewnętrznej. Uwaga: nie wszystkie usługi wewnętrzne są usługami sieciowymi, więc ten identyfikator nie zawsze będzie mieć postać adresu URL. Przy weryfikacji biletu pośredniego tego typu identyfikator musi być jednak zgodny z parametrem „service" wprowadzonym dla funkcji /proxyValidate.

2.7.2. Odpowiedź

Funkcja /proxy zwróci odpowiedź usługi CAS („CAS serviceResponse") w formacie XML, zgodnie z opisem struktury XML w załączniku A. Poniżej przedstawiono przykładowe odpowiedzi:

W przypadku udanego żądania:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
    <cas:proxySuccess>
        <cas:proxyTicket>PT-1856392-b98xZrQN4p90ASrw96c8</cas:proxyTicket
>
    </cas:proxySuccess>
</cas:serviceResponse>
```

W przypadku nieudanego żądania:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
    <cas:proxyFailure code="INVALID_REQUEST">
        'pgt' and 'targetService' parameters are both required
    </cas:proxyFailure>
</cas:serviceResponse>
```

2.7.3. Kody błędów

Poniższe wartości mogą zostać użyte jako atrybuty „kodowe" odpowiedzi informujących o nieudanej weryfikacji. Jest to minimalna lista kodów błędów, które muszą obsługiwać wszystkie serwery CAS. Poszczególne implementacje mogą obsługiwać też kody innego rodzaju.

- INVALID_REQUEST – nie wszystkie wymagane parametry żądania są dostępne.
- BAD_PGT – wprowadzono nieprawidłowy pgt.
- INTERNAL_ERROR – podczas weryfikacji biletu wystąpił błąd wewnętrzny.

W przypadku wszystkich kodów błędów zaleca się, by w głównej części sekcji <cas:authenticationFailure> w odpowiedzi XML system CAS zamieścił szczegółowe informacje na ich temat.

2.7.4. Przykładowy adres URL dla funkcji /proxy

Przykładowe żądanie dot. pośredniczenia:

```
https://server/cas/proxy?targetService=http%3A%2F%2Fwww.service.com&pgt=.
....
```

3. Obiekty CAS

3.1. Bilet usługi

Bilet usługi to nieprzejrzysty ciąg znaków, którego klient używa jako danych identyfikacyjnych w celu uzyskania dostępu do usługi. Bilet usługi pozyskiwany jest z systemu CAS po wprowadzeniu przez klienta do funkcji /login danych identyfikacyjnych oraz identyfikatora usługi, zgodnie z opisem w rozdziale 2.2.

3.1.1. Właściwości biletu usługi

- Bilety usług ważne są wyłącznie dla identyfikatora usług ustanowionego dla funkcji /login w momencie ich generowania. Identyfikator usługi nie powinien stanowić części biletu usługi.
- Bilety usług muszą być ważne wyłącznie dla jednej próby weryfikacji biletu. Niezależnie od tego, czy weryfikacja się powiodła, system CAS musi unieważnić bilet, tym samym uniemożliwiając kolejne próby jego weryfikacji.

- System CAS powinien spowodować unieważnienie biletów, których weryfikacja się nie powiodła po upływie określonego czasu od ich wydania. Jeśli usługa przedłoży do weryfikacji bilet usługi, którego termin ważności upłynął, system CAS musi odpowiedzieć komunikatem o nieudanej weryfikacji. Zaleca się, by komunikat taki zawierał opis przyczyn nieudanej weryfikacji. Zaleca się także, by okres ważności biletu usługi wynosił nie dłużej niż pięć minut. Optymalny czas na wykorzystanie biletów niepoddanych weryfikacji mogą regulować lokalne zasady bezpieczeństwa i użytkowania systemu CAS.
- Ze względów bezpieczeństwa bilety usług muszą zawierać odpowiednio rozłożone przypadkowe dane, tak aby treści biletu nie dało się łatwo odczytać.
- Bilety usług muszą rozpoczynać się ciągiem znaków „ST-„.
- Usługi muszą obsługiwać bilety usług o maks. długości 32 znaków. Zaleca się, by usługi obsługiwały bilety usług o maks. długości 256 znaków.

3.2. Bilet pośredni

Bilet pośredni to nieprzezroczysty ciąg znaków stosowany przez usługę w imieniu klienta do uzyskania dostępu do usługi wewnętrznej. Bilety pośrednie uzyskiwane są od systemu CAS po przedstawieniu usłudze ważnego biletu nadania pośrednictwa (rozdział 3.3) oraz identyfikatora usługi dla usługi wewnętrznej, do której się podłącza.

3.2.1. Właściwości biletów pośrednich

- Bilety pośrednie ważne są wyłącznie dla identyfikatora usług ustanowionego dla funkcji /proxy w momencie ich generowania. Identyfikator usługi nie powinien stanowić części biletu usługi.
- Bilety pośrednie muszą być ważne wyłącznie dla jednej próby weryfikacji biletu. Niezależnie od tego, czy weryfikacja się powiodła, system CAS musi unieważnić bilet, tym samym uniemożliwiając kolejne próby jego weryfikacji.
- System CAS powinien spowodować unieważnienie biletów, których weryfikacja się nie powiodła po upływie odpowiednio dużej ilości czasu od ich wydania. Jeśli usługa przedłoży do weryfikacji bilet pośredni, którego termin ważności upłynął, system CAS musi odpowiedzieć komunikatem o nieudanej weryfikacji. Zaleca się, by komunikat taki zawierał opis przyczyn nieudanej weryfikacji. Zaleca się także, by okres ważności biletu pośredniego wynosił nie dłużej niż pięć minut. Optymalny czas na wykorzystanie biletów niepoddanych weryfikacji mogą regulować lokalne zasady bezpieczeństwa i użytkowania systemu CAS.
- Ze względów bezpieczeństwa, bilety pośrednie muszą zawierać odpowiednio rozłożone przypadkowe dane, tak aby treści biletu nie dało się łatwo odczytać.
- Bilety pośrednie muszą rozpoczynać się ciągiem znaków „PT-„.

- Usługi muszą obsługiwać bilety pośrednie o maks. długości 32 znaków. Zaleca się, by usługi obsługiwały bilety pośrednie o maks. długości 256 znaków.

3.3. Bilet nadania pośrednictwa

Bilet nadania pośrednictwa to nieprzezroczysty ciąg znaków stosowany przez usługę w imieniu klienta do uzyskania dostępu do usługi wewnętrznej. Bilety nadania pośrednictwa pozyskiwane są z systemu CAS po weryfikacji biletu usługi lub biletu pośredniego. Procedurę wydania biletu nadania pośrednictwa opisano szczegółowo w rozdziale 2.5.4.

3.3.1. Właściwości biletu nadania pośrednictwa

- Bilety nadające pośrednictwo mogą być wykorzystane przez usługi w celu uzyskania wielu biletów pośrednich. Bilety nadania pośrednictwa nie są przeznaczone do jednokrotnego użytku.
- Termin ważności biletów nadania pośrednictwa musi upłynąć w momencie wylogowania się z systemu CAS klienta, którego dane uwierzytelniające przesyłane są poprzez pośrednika.
- Bilety nadania pośrednictwa muszą zawierać zabezpieczone, przypadkowe dane, tak aby odpowiednio zapobiec możliwości ich odczytania przez atak typu „brute force".
- Bilety nadania pośrednictwa powinny rozpoczynać się ciągiem znaków „PGT-".
- Usługi muszą obsługiwać bilety nadania pośrednictwa o maks. długości 64 znaków. Zaleca się, by usługi obsługiwały bilety nadania pośrednictwa o maks. długości 256 znaków.

3.4. IOU biletu nadania pośrednictwa

IOU biletu nadania pośrednictwa to nieprzezroczysty ciąg znaków umieszczony w odpowiedzi funkcji /serviceValidate i /proxyValidate. Stosuje się go do skorelowania biletu usługi lub weryfikacji biletu pośredniego z danym biletem nadania pośrednictwa. Pełen opis tej procedury zamieszczono w dziale 2.5.4.

3.4.1. Właściwości IOU biletu nadania pośrednictwa

IOU biletów nadania pośrednictwa (PGTIOU) nie powinny zawierać żadnych odniesień do powiązanych z nimi biletów nadania pośrednictwa. Należy zapobiec możliwości odczytania z danego PGTIOU, w oparciu o metody algorytmiczne i w odpowiednio długim przedziale czasu, informacji o tym, do jakiego biletu nadania pośrednictwa się on odnosi.

- IOU biletów nadania pośrednictwa muszą zawierać zabezpieczone, przypadkowe dane, tak aby odpowiednio długo zapobiec możliwości odczytania biletu przez atak typu „brute force".
- IOU biletów nadania pośrednictwa powinny rozpoczynać się ciągiem znaków „PGTIOU-".
- Usługi muszą obsługiwać PGTIOU o maks. długości 64 znaków. Zaleca się, by usługi obsługiwały PGTIOU o maks. długości 256 znaków.

3.5. Bilet logowania

Bilet logowania to ciąg znaków udostępniany przez funkcję /login jako żądającego danych identyfikacyjnych i przekazywany do tej funkcji jako przyjmujący dane identyfikacyjne podczas uwierzytelniania nazwy użytkownika / hasła. Bilet ten ma na celu przede wszystkim zapobiec konieczności ponownego wprowadzania danych wskutek błędów w przeglądarkach internetowych.

3.5.1. Właściwości biletu logowania

- Bilety logowania wydawane przez funkcję /login muszą być w jak największym stopniu niepowtarzalne.
- Bilety logowania muszą być ważne wyłącznie dla jednej próby weryfikacji biletu. Niezależnie od tego, czy weryfikacja się powiodła, system CAS musi unieważnić bilet, tym samym uniemożliwiając kolejne próby jego weryfikacji.
- Bilety logowania powinny rozpoczynać się ciągiem znaków „LT-".

3.6. Plik cookie udostępniający bilet

Plik cookie to plik cookie typu HTTP[5] tworzony przez system CAS po zainicjowaniu sesji pojedynczego logowania. Plik ten zawiera informacje o statusie logowania klienta i dopóki jest ważny, klient może posługiwać się nim w systemie CAS zamiast podstawowych danych identyfikacyjnych. Usługi mogą zostać wycofane z sesji pojedynczego logowania za pomocą parametru „renew" opisanego w rozdziałach 2.1.1, 2.4.1 i 2.5.1.

3.6.1. Właściwości pliku cookie udostępniającego bilet

- Plik cookie udostępniający bilet musi być skonfigurowany tak, aby tracił ważność wraz z końcem sesji przeglądarki klienta.
- Ścieżka dostępu do pliku cookie w systemie CAS musi być ściśle określona. Jeśli na przykład serwer CAS ustawiono do obsługi ścieżki /cas, ścieżka dostępu do pliku cookie również musi być ustawiona na /cas.

- Wartość pliku cookie udostępniającego bilet musi zawierać zabezpieczone przypadkowe dane, tak aby uniemożliwić odczytanie tego pliku przez odpowiednio długi okres.
- Wartość pliku cookie udostępniającego bilet powinna rozpoczynać się ciągiem znaków „TGC-".

3.7. Zbiór znaków w biletach i plikach cookie udostępniających bilety

Dodatkowym wymogiem jest to, by wszystkie bilety CAS oraz wartości plików cookie udostępniających bilety obowiązkowo zawierały znaki ze zbiorów: {A-Z, a-z i 0-9 oraz znak łącznika}.

Załącznik A: Schemat XML odpowiedzi CAS

```
<!--
    The following is the schema for the Yale Central Authentication
    Service (CAS) version 2.0 protocol response.  This covers the responses
    for the following servlets:
      /serviceValidate
      /proxyValidate
      /proxy
    This specification is subject to change.
    Author:  Drew Mazurek
    Version: $Id: cas2.xsd,v 1.1 2005/02/14 16:19:06 dmazurek Exp $
-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:cas="http://www.yale.edu/tp/cas"
           targetNamespace="http://www.yale.edu/tp/cas"
          elementFormDefault="qualified" attributeFormDefault="unqualified">
    <xs:element name="serviceResponse" type="cas:ServiceResponseType"/>
    <xs:complexType name="ServiceResponseType">
        <xs:choice>
            <xs:element name="authenticationSuccess"
type="cas:AuthenticationSuccessType"/>
```

```
            <xs:element name="authenticationFailure"
type="cas:AuthenticationFailureType"/>
            <xs:element name="proxySuccess" type="cas:ProxySuccessType"/>
            <xs:element name="proxyFailure" type="cas:ProxyFailureType"/>
        </xs:choice>
    </xs:complexType>
    <xs:complexType name="AuthenticationSuccessType">
        <xs:sequence>
            <xs:element name="user" type="xs:string"/>
            <xs:element name="proxyGrantingTicket" type="xs:string"
minOccurs="0"/>
            <xs:element name="proxies" type="cas:ProxiesType" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ProxiesType">
        <xs:sequence>
            <xs:element name="proxy" type="xs:string" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="AuthenticationFailureType">
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="code" type="xs:string" use="required"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
    <xs:complexType name="ProxySuccessType">
        <xs:sequence>
            <xs:element name="proxyTicket" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>
```

```
    <xs:complexType name="ProxyFailureType">
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="code" type="xs:string" use="required"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
</xs:schema>
```

Załącznik B: Bezpiecznie przekierowanie

Po pomyślnym zalogowaniu bezpieczne przekierowanie klienta z CAS do miejsca docelowego musi odbywać się z zachowaniem należytej ostrożności. W większości przypadków klient wysłał już dane identyfikacyjne do serwera CAS w żądaniu POST. Zgodnie ze specyfikacją serwer CAS musi następnie odesłać użytkownika do aplikacji za pomocą żądania GET.

HTTP/ 1.1 RFC[3] dysponuje kodem odpowiedzi „303: See Other" („303: Sprawdź inne"), który odpowiada za właściwe działanie: skrypt otrzymujący dane na podstawie żądania POST może, poprzez przekierowanie 303, przekazać przeglądarce inny adres URL w oparciu o żądanie GET. Należy jednak pamiętać, że nie wszystkie przeglądarki potrafią poprawnie obsłużyć ten tryb działania.

W związku z tym zalecaną metodą przekierowania jest JavaScript. Strona zawierająca własność window.location.href zbudowaną według poniższego wzoru działa następująco:

```
<html>

    <head>
        <title>Yale Central Authentication Service</title>
        <script>
            window.location.href="https://portal.yale.edu/Login?ticket=ST-..."
mce_href="https://portal.yale.edu/Login?ticket=ST-...";
```

```
        </script>

    </head>
    <body>
         <noscript>
             <p>CAS login successful.</p>
             <p>  Click <a xhref="https://portal.yale.edu/Login?ticket=ST-..."
mce_href="https://portal.yale.edu/Login?ticket=ST-...">here</a>
             to access the service you requested.<br />  </p>
         </noscript>
     </body>
</html>
```

Ponadto system CAS powinien wyłączyć buforowanie pamięci podręcznej przeglądarki, ustawiając nagłówki związane z pamięcią podręczną:

- Pragma: no-cache
- Cache-Control: no-store
- Expires („wygaśnięcie"): [RFC 1123[6] data aktualna lub wcześniejsza]

Wprowadzenie biletu logowania uniemożliwia CAS przyjmowanie danych identyfikacyjnych, które zostały zapisane w pamięci podręcznej i przez nią powtórzone. Jednak wczesne wersje przeglądarki Apple Safari zwierały błąd, który polegał na tym, że po kliknięciu przycisku „Wstecz" przeglądarka mogła zostać zmuszona do przedstawienia danych identyfikacyjnych klienta usłudze, do której chciała uzyskać dostęp. System CAS może zapobiec takiemu zachowaniu uniemożliwiając automatyczne przekierowanie, jeżeli wykryje, że zdalna przeglądarka jest jedną z wczesnych wersji Safari. W zamian CAS powinien wyświetlić stronę z informacją o pomyślnym zalogowaniu oraz odnośnik do żądanej usługi. Aby przejść dalej, klient musi samodzielnie kliknąć ten odnośnik.

Załącznik C: Bibliografia

[1] Bradner, S., „Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, Harvard University, marzec 1997.

[2] Berners-Lee, T., Fielding, R., Frystyk, H., „Hypertext Transfer Protocol - HTTP/1.0", RFC 1945, MIT/LCS, UC Irvine, MIT/LCS, maj 1996.

[3] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T., „Hypertext Transfer Protocol - HTTP/1.1", RFC 2068, UC Irvine, Compaq/W3C, Compaq, W3C/MIT, Xerox, Microsoft, W3C/MIT, czerwiec 1999.

[4] Berners-Lee, T., Masinter, L. i MaCahill, M., „Uniform Resource Locators (URL)", RFC 1738, CERN, Xerox Corporation, University of Minnesota, grudzień 1994.

[5] Kristol, D., Montulli, L., „HTTP State Management Mechanism", RFC 2965, Bell Laboratories/Lucent Technologies, Epinions.com, Inc., październik 2000.

[6] Braden, R., „Requirements for Internet Hosts - Application and Support", RFC 1123, Internet Engineering Task Force, październik 1989.

Załącznik D: Licencja CAS

Niniejsze oprogramowanie jest dostarczane „jak jest" (as is), a wszelkie wyraźne i dorozumiane gwarancje, a w szczególności dorozumiane gwarancje wartości handlowej lub przydatności użytkowej zostają jednoznacznie wyłączone. W żadnym wypadku Uniwersytet Yale ani jego pracownicy nie ponoszą odpowiedzialności za jakiekolwiek bezpośrednie, pośrednie, uboczne, specjalne, prewencyjne lub wynikowe szkody (a w szczególności koszty zaopatrzenia w zamienne towary i usługi; utratę użyteczności, danych lub zysków; lub przerwy w działalności), bez względu na ich przyczynę, podlegające jakimkolwiek podstawom zobowiązania, z tytułu odpowiedzialności umownej, deliktowej bądź obiektywnej (w tym dotyczącej zaniedbań lub innych), wynikłe w dowolny sposób z użytkowania oprogramowania, nawet po uprzednim poinformowaniu o możliwości wystąpienia takich szkód.

Redystrybucja i użytkowanie niniejszego oprogramowania w formie źródłowej lub binarnej, z modyfikacjami lub bez, jest dozwolone pod warunkiem spełnienia następujących warunków:

1. Do każdej redystrybucji należy dołączyć powyższą informację o prawach autorskich oraz klauzulę zrzeczenia się odpowiedzialności, a także niniejszą listę warunków w treści powiązanej dokumentacji i, o ile to możliwe, w redystrybuowanym oprogramowaniu.

2. Każda redystrybucja musi zawierać informacje o źródle w formie: „Niniejszy produkt zawiera oprogramowanie opracowane prze Uniwersytet Yale" w treści powiązanej dokumentacji i, o ile to możliwe, w redystrybuowanym oprogramowaniu.

3. Nazwy „Yale" i „Uniwersytet Yale" nie będą używane w celu wsparcia lub promocji produktów opracowanych w oparciu o niniejsze oprogramowanie.

Załącznik E: Zmiany

4 maja 2004 r. – wydanie pierwsze

2 maja 2012 r.: wersja 1.0.1 – poprawiono literówkę w „noscropt". apetro per amazurek z podziękowaniami dla Faraza Khana z ASU za znalezienie błędu.