HTTP Digest Kimliklendirme

Digest Authentication Nedir?

- Basic ve Form Authentication'ın **HTTP ortamındaki zaafiyetleri**ni gidermek için geliştirilmiş yöntemdir
- Temel amacı HTTP ortamında **credential bilgileri**nin kolayca çözümlenmesinin veya clear text formatta transfer edilmesinin önüne geçmektir
- **HTTPS'in kullanılamadığı durumlar için** form ve basic authentication'dan çok daha güvenlidir

Digest Authentication Nedir?

- **Pek çok tarayıcı** tarafından desteklenir
- RFC 2069 ve RFC 2617 tarafından tanımlanır
- Spring Security gerçekleştirimi bu RFC'lerle uyumludur

Digest Authentication İsteğinin Yapısı

- Digest authentication sunucunun aşağıdaki yapıda **401 statü kodlu bir HTTP cevabı** dönmesi ile başlar

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest realm="<realm>",
                  domain="<domain>",
                  nonce="<nonce>",
                  opaque="<opaque>",
                  stale="<TRUE | FALSE>"
```

Tarayıcı tarafından görüntülenecek login dialog penceresinde görüntülenen kullanıcının neresi için username, password gireceğini anlamasını sağlayan bir mesajdır

Tarayıcının bu auth bilgisini hangi URI'lar için gönderdiğini bilmesini sağlayacak opsiyonel bir bilgidir, gönderilmez ise auth bilgisi bütün URI'lar için gönderilmiş kabul edilir

Sunucu tarafından her 401 cevabı için Benzersiz biçimde yeniden üretilen reply saldırılarının önüne geçmeyi sağlayan istemci için anlamı olmayan integer bir değerdir

İstemci tarafından hiç değiştirilmeden geri gönderilecek Base64 veya hex bir String değerdir

Bir önceki isteğin geçersiz bir nonce değeri nedeni ile reddedildiğini belirten bir flag'tir

Digest Authentication İsteğinin Yapısı

- İstemci tarafından gönderilecek HTTP isteğinde yer alması gereken **Authorization header değeri** de aşağıdaki gibi bir yapıya sahiptir

```
Authorization: Digest
            username="<username>",
            realm="<realm>",
            nonce="<nonce>",
            uri="<requested-uri>",
            response="<digest>",
            message="<message-digest>",
            opaque="<opaque>"
```

Opsiyoneldir http isteğinin içeriğinin Değişip değişmediğini kontrol amaçlı kullanılır

```
<digest> := H( H(A1) + ":" + N + ":" + H(A2) )
<message-digest> := H( H(A1) + ":" + N + ":" + H(<http-message-body>) )
A1 := U + ':' + R + ':' + P
A2 := <http-method> + ':' + <requested-uri>
```

Digest Authentication İsteğinin İşlenmesi

- Authorization request header'ı sunucu tarafına ulaşınca sunucu belirtilen **username'e karlışılık gelen password'ü DB'den elde edip** istemci tarafından gerçekleştirilen MD5 hash işlemini tekrarlayarak **<response> digest değerini** elde eder ve bunu istemciden gelen değer ile karşılaştırır

Digest Authentication İsteğinin İşlenmesi

- Başarılı kimliklendirme sonrasında sunucu istemciye **opsiyonel olarak** aşağıdaki gibi bir cevap dönebilir

```
HTTP/1.1 200 OK
Digest-MessageDigest:
                username="<username>",
                realm="<realm>",
                nonce="<nonce>",
                message="<message-digest>"
```

Sunucu tarafından http isteğinin içeriğiden üretilen digest değeridir, böylece istemci gönderdiği içeriğin değişmeden sunucuya ulaştığından emin olabilir

Digest Authentication Konfigürasyonu

```
<bean id="digestAuthenticationFilter"
class="org.springframework.security.web.authentication.www.DigestAuthenticationFilter">

   <property name="userDetailsService"
ref="userService" />

   <property name="authenticationEntryPoint"
ref="digestAuthenticationEntryPoint" />

</bean>
```

DigestAuthenticationFilter kullanıcının **clear text password**'üne erişebilmelidir

Digest Authentication Konfigürasyonu

```
<bean id="digestAuthenticationEntryPoint"
class="org.springframework.security.web.authentication.www.DigestAuthenticationEntryPoint">

   <property name="realmName" value="Digest Realm" />

   <property name="key" value="secret" />

   <property name="nonceValiditySeconds" value="10" />

</bean>
```

Digest Authentication Konfigürasyonu

```
<security:http pattern="/mvc/**"
   create-session="stateless"
   entry-point-ref = "digestAuthenticationEntryPoint">

   <security:custom-filter
           ref="digestAuthenticationFilter"
           after="BASIC_AUTH_FILTER"/>

   <security:intercept-url pattern="/**"
           access="isFullyAuthenticated()" />

</security:http>
```

Fonksiyonel olarak BasicAuthenticationFilter ile aynı role sahip olduğu için bu filter'dan önce veya sonra yerleştirilebilir

DigestAuthenticationFilter

- BasicAuthenticationFilter gibi **akışı kesintiye uğratmadan** devreye girer
- HTTP request header'ındaki **Digest Authorization Request Header**'ını işleyip geçerli bir **UsernamePasswordAuthenticationToken** oluşturur
- Bu authentication token'da doğrudan **SecurityContext**'e set edilir
- Böylece authentication işlemi **AuthenticationManager** devreye girmeden halledilmiş olur

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com