Kerberos in Your JVM

An Introduction to Apache Kerby

Kiran Ayyagari
kayyagari@apache.org
Kiran Ayyagari

- Apache Member
- Chairman of Apache Directory Project
- Involved with ApacheDS since 2008
- Independent Consultant
What is Kerberos?

• An authentication protocol
• Designed to work over untrusted networks
• Passwords are NOT sent over wire
What is Kerberos?

- A classical Single SignOn solution
- Authorization at OS host level
How Does it Work?
Participants

Alice

Authentication Server + TicketGranting Server

Kerberos Server

Bob
General Payload Structure

General Request/Response Payload

- Data in plain text
- Ticket
- Encrypted Data

Ticket

- Data in plain text
- Encrypted Data
Part - I

Auth Request for a Ticket Granting Ticket

Alice

Authentication Server

Session Key 1 along with a Ticket Granting Ticket
Part - II

Request for a Ticket to communicate with Bob

Alice

Session Key 2 along with a Ticket to communicate with Bob

Ticket Granting Server
Part - III

Request with the session Ticket to Bob

(Optional) Response to verify that it is Bob indeed (mutual verification)
Where it is Used?

In authenticating users

• on workstations

• in network services like SSH, FTP

and in Apache Hadoop
Kerberos at Apache?

- Part of ApacheDS since 2004
- Written by one person, Enrique Rodriguez
- Tightly coupled with LDAP backend
- Totally reviewed in 2010
- Client implementation was added in 2011
ApacheDS Kerberos Status?

• Functional
• Tightly coupled with Directory Server
• Lacks many features (cross-realm, pkinit, FAST etc..)
• Complex codebase
• Not enough maintainers
What’s Next?

• Zheng Kai from Intel started working on a simplified codec

• Proposed to bring his effort to Apache Directory Project

• Jiajia Li, Lin Chen and Xu Yanning, all from Intel joined the effort

• Development was apace and resulted in release of a fully functional server and client with several features
Apache Kerby

• http://directory.apache.org/kerby

• A Kerberos v5 server written in java

• Can run standalone or in-process

• supports transient and numerous persistent storage options

• Bundled with a client, kadmin and other utilities

• An excellent choice for unit testing kerberized clients and servers
Embedding Kerby

KdcServer kdc = new KdcServer();
NettyKdcServerImpl network = new NettyKdcServerImpl(kdc.getKdcSetting());

kdc.setXXX(); // set the basic settings, host, port, protocol and realm
kdc.init();
kdc.start();

kadmin = new Kadmin( kdc.getKdcSetting(), kdc.getIdentityService() );
kadmin.createBuiltinPrincipals();
kadmin.addPrincipal( "elecharny", "sha1024" );

// THAT IS ALL ;)
Kerby in Unit Tests

@BeforeClass
public static void setup() throws Exception {
    // start KDC
    // initialize client
}

@AfterClass
public static void stop() throws Exception {
    // stop KDC
}

@Test
public void testGetTGTticket() throws Exception {
    TgtTicket tgt = client.requestTgtWithPassword("el@EXAMPLE.COM", "secret");
    assertNotNull(tgt);
}
Using Kerberos over HTTP?

• SPNEGO works but won’t work out of box everywhere

• JWT seems promising
JSON Web Token

- A compact URL-safe means of representing claims to be transferred between two parties
- Contains a Header, Claims and Signature
  <header>.<claims>.<signature>
- All parts are Base64 encoded individually
- Header: {"typ":"JWT", "alg":"HS256"}
- Claims: {"iss":"elecharny", "exp":1300819380}
Example App

https://github.com/kayyagari/krb2jwt
Kerberos Ticket to JWT

JWT Header:
{
    "srvtkt": <base64-encoded-Ticket>,
    "keytype": "aes128-cts-hmac-sha1-96",
    "alg": "HS512"
}

JWT Claims:
{
    "aud": "webapp1@EXAMPLE.COM",
    "exp": "1443706562444",
    "iat": "1443706262444",
    "iss": " krb2jwt",
    "sub": "elecharny@EXAMPLE.COM"
}
Usecases of Krb2JWT

- HTTP clients communicating via backchannel
- Hadoop nodes
Roadmap

- PKINIT
- Cross-Realm
- OTP based ticket granting
Questions?

http://directory.apache.org/kerby
Thank You!

Zheng Kai and his band at Intel

Emmanuel Lecharny (elecharny@apache.org)

Stefan Seelmann (seelmann@apache.org)