Interrealm Authentication in Kerberos Version 5

Somesh Jha
Computer Sciences Department
University of Wisconsin
Madison, WI 53706.

April 8, 2003

Scenario: Assume that user U is in realm R_1 and wants to access the server V in realm R_k. There is a path $R_1 \rightarrow R_2 \rightarrow \cdots \rightarrow R_k$ from realm R_1 to R_k. Conceptually, each edge $R_i \rightarrow R_{i+1}$ (for $1 \leq i < k$) represents a trust relationship between realm R_i and R_{i+1}, which usually means that there is a shared key between the two realms.
Initial request: U requests a *ticket-granting ticket* or *TGT* from the KDC in realm R_1 (which we denote by $KDC[R_1]$) for realm R_k with the `FORWARDABLE` flag on.[1] Since R_1 does not have a trust relationship with R_k, it issues a TGT $TGT[R_1 \rightarrow R_2]$ for realm R_2 with the `FORWARDABLE` flag on. We are assuming that there is a mechanism for realm R_1 to discover that there is a path to realm R_k that goes through R_2. *Note:* I am also assuming that the servers only issue these tickets if their policy allows it. For example, $KDC[R_1]$ only issues the TGT with the `FORWARDABLE` flag on to U, if its policy allows it. This will be implicit throughout the document.
Walking the path: Using the TGT $TGT[R_1 \rightarrow R_2]$, U requests a TGT for realm R_3 from the *ticket granting server* or *TGS* (denoted by $TGS[R_2]$) in realm R_2. The TGT issued by $TGS[R_2]$ (denoted by $TGT[R_2 \rightarrow R_3]$) for R_3 has the `FORWARDABLE` and `FORWARDED` flags on. The $TGT[R_2 \rightarrow R_3]$ can have a different address than U (presumably an agent is handling this on behalf of the user U). This process is repeated until U "reaches" the realm R_k, i.e., it has a TGT $TGT[R_{k-1} \rightarrow R_k]$ issued by $TGS[R_{k-1}]$ for the realm R_k.
Accessing V: The TGT $TGT[R_{k-1} \rightarrow R_k]$ is presented to the TGS $TGS[R_k]$ to obtain a *service-granting ticket* or *SGS* $SGT[R_k, V]$ for server V. This SGS can then be used to access the server V.

0.1 Critique of Interrealm Authentication in Kerberos

This subsection describes some of the shortcomings of interrealm authentication in Kerberos.
Implicit Trust Relationships: There are implicit trust relationships between realms, which in the Kerberos context manifests as sharing keys between realms. If a realm R_i issues a TGT for realm R_j, it abstractly denotes that R_j is trusting R_i for authenticating the user. We would like to make these trust relationships explicit.

[1] In general, an entity will be indexed by the realm that it pertains to, e.g., a ticket granting ticket or TGT issued by realm R_i for realm R_j will be denoted by $TGT[R_i \rightarrow R_j]$.

Closed-world Assumption: Imagine that there are two realms in an university. The two realms, R_B and R_C, correspond to the biology and the computer science department respectively. Let us say that a professor A in the biology department wants to provide access to a server V to all group members that belong to the project *cloneSheep* in computer science. In the current scheme, A will have to know all the group members of the project. In other words, professor A has to know the identity of all the group members of *cloneSheep*, which violates the closed-world assumption. Ideally, professor A should be able to specify that *server V is accessible to all group members of cloneSheep* and authorization should happen seamlessly. This also has the advantage that if the group changes (for example, a member leaves), the authorization decision should seamlessly incorporate this new information (without A having to explicit change ACLs).

We claim that by using trust management in conjunction with a distributed authentication service, such as Kerberos, we can address the two shortcomings described above.