Application mobility with Host Identity Protocol – Extended Abstract

Teemu Koponen and Andrei Gurtov
Helsinki Institute for Information Technology
{tkoponen,gurtov}@hiit.fi

Pekka Nikander
Ericsson Research NomadicLab
pekka.nikander@nomadiclab.com

Introduction

In this paper, we consider process migration from a communications point of view. We use the term *application mobility* while referring to an application being moved from a host to another during its execution. In this paper these hosts are called *source* and *destination* hosts, respectively. Moreover, we define a host to be virtual, and thus, not to equal to a physical host as such. Therefore, multiple (virtual) hosts may coexist within a physical host.

The application movement should be transparent for both applications themselves and hosts they are communicating with. Neither it should compromise the security of communications. These requirements form the problem we are responding to. We enhance Host Identity Protocol (HIP)[3] to meet the requirements, because we see support for multiple Host Identities (HI) in a physical host as logical development for the HIP architecture.

HIP introduces a new layer into the TCP/IP protocol stack between the transport and network layers. In essence, the HIP layer is an isolator between applications and mobility. As we include the state of the communication stack upwards from the transport layer to the migrated application state, we can migrate applications regardless of the transport and application protocols they use. The three fundamental challenges which must be addressed to provide such transparency for the transport layer, applications, and connected hosts are managing state inconsistencies, resolving state conflicts, and maintaining state syncronization[4]:

Preserving connection states is difficult as the transport protocols are not designed with (application) mobility in mind, and an address change, caused by mobility, is subject to introduce *state inconsistencies* between the network layer and transport layer.

Application mobility may create *state conflicts* in the transport layer. If an address is relocated together with an application to a destination host and the same address were used again in the source host, it would be possible to have connections from two hosts to a third host with identical connection invariants.

Introduction of NAT (Network Address Translation) and NAPT (Network Address Port Translation) devices make *state synchronization* between hosts challenging as the same address:port–address:port tuple is no more usable as a connection invariant in the both communication end-points.

The above challenges are the first concrete signs for us to imply that HIP might provide an elegant solution to the problem of providing transparency for the transport layer, applications and connected hosts in an application mobility architecture. To elaborate, as HIP separates the transport layer from mobility induced network address changes, HIP can offer a cure for the first two challenges. As NAT and NAPT devices do not intermix with host identities, the third challenge becomes also manageable.

Delegation – the missing piece

One could simply consider application mobility as a form of mobility or multi-homing. However, the situation is more complex, as in HIP communicating a multi-homing or mobility caused address change to other hosts requires a host to have an access to a private cryptographic key of its host identity[3]. Therefore, the private key should be moved, together with an application, from a host to another to maintain the host's multi-homing and mobility abilities. While moving a cryptographic key may be impossible in cases where secure tamper proof physical storages are used, it could also result in undesirable security implications. Moreover, as HIP host identities are architecturally more bound to hosts than to applications, the identities should not be moved.

Unfortunately, if a private cryptographic key can't be transferred, a destination host of a migrated application becomes dependent on a source host. Thus, we would introduce a strong *residual dependency*[1] to the source host. Clearly, the host identity must change if an application moves from a host to another. However, currently the host identities are assumed to remain the same for the whole lifetime of a HIP association. Therefore, we now consider the implications if HIP had support for changing host identities transparently from the transport layer's and other hosts' point of view.

A Host Identity Tag (HIT) is a cryptographic hash of a host identity[3]. As the transport layer and applications bind to HITs and not to host identities, instead of moving a host identity of a source host, the source host *delegates* the responsibility of its HIT to a destination host. As a result, hosts can assume the destination host to represent the HIT of the source host, even the host identity is dif-

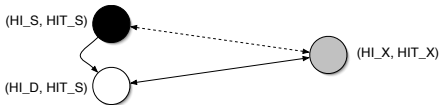

Figure 1: A delegation certificate (a curve) authorizes a destination host (a hollow circle) to use the HIT of a source host (a black circle) for the communication with another host (a grey circle).

ferent. In the following, we consider the realization of the missing piece, namely the delegation, in more detail.

Establishing and maintaining associations

In basic HIP, DNS stores the contact and host identity information of hosts. However, it is difficult to keep DNS up to date with the application mobility caused host identity changes. Thus, a destination host must be able to respond to HIP association handshakes directed to a HIT of a source host. Moreover, it must prove its authorization to the specific HIT whether the source host is available or not. Therefore, the source host must sign a *delegation certificate* with the private cryptographic key of its host identity. The certificate enables a host with the source host identity information to validate the destination's authority to the source HIT, as the certificate binds the old source host identity information (still in DNS) to the destination host identity (not yet in DNS). Figure 1 depicts the concept of delegation.

Applications may migrate multiple times sequentially, even in a rapid manner, and thus, DNS may contain host identity information older than the last application movement would let assume. Therefore, we do not restrict the number of delegation certificates a destination host may include to handshake responses. Instead, hosts can construct a trust chain from the received certificates.

Finally, to maintain the plain mobility and multi-homing abilities, a destination host must inform a respective HIP rendezvous server about its address changes. To prove its authority to a source HIT, the destination host includes the necessary delegation certificates while contacting the server to update its current address. Updating already established HIP associations to reflect the address changes is simple, as the destination's authority to the source HIT was already validated while establishing the HIP associations.

Transparency for applications

Actions of an application have established HIP associations to hosts while it runs within a source host. Thus, before the application may move, the destination host must establish new HIP associations to the same hosts. As when updating the rendezvous information, it is enough that the destination host merely includes necessary delegation certificates to handshakes.

There could be several HIP associations established with the same HIT as it's represented simultaneously by both the source and the destination host. However, as two hosts are expected to share a single HIP association, the new associations, the destination host establishes, replace the old ones. This ultimately renders application mobility transparent for the layers above HIP within the hosts that applications communicates with.

Maintaining transparency for the application within the destination host is straight-forward. Establishment of HIP associations together with the delegation certificates already introduced the delegated source HIT, as a side-effect, for the layers above HIP.

Conclusions

In this paper, we have suggested a new application mobility architecture based on HIP and delegation certificates. The usability of the delegation certificates is by no means limited to application mobility; instead, a host may certify another host to impersonate itself, even though an application does not migrate. Thus, our architecture resembles the Delegation-Oriented Architecture (DOA)[5]. Clearly, further work is required to understand the wider architectural implications of the new proposed delegation primitive and the relationship to signaling delegation in general[2].

Our initial work has already shown the importance of truly supporting multiple host identities within a physical host. Finally, we note that the delegation primitive enhances HIP to actually support two layers of mobility in a secure way: mobility due to physical movement and mobility due to management and administrative causes.

References

[1] D. S. Milojičić, F. Douglis, Y. Paindaveine, R. Wheeler, and S. Zhou. Process Migration. *ACM Computing Surveys*, 32(3):241–299, 2000.

[2] P. Nikander and J. Arkko. Delegation of Signalling Rights. In *Proceedings of Security Protocols, 10th International Workshop*, pages 203–212, Cambridge, UK, Apr. 2002. Springer.

[3] P. Nikander, J. Ylitalo, and J. Wall. Integrating Security, Mobility, and Multi-Homing in a HIP Way. In *Proceedings of Network and Distributed Systems Security Symposium (NDSS'03)*, pages 87–99, San Diego, CA, USA, Feb. 2003. Internet Society.

[4] G. Su. *MOVE: Mobility with Persistent Network Connections*. PhD thesis, Columbia University, Oct. 2004.

[5] M. Walfish, J. Stribling, M. Krohn, H. Balakrishnan, R. Morris, and S. Shenker. Middleboxes No Longer Considered Harmful. In *Proceedings of the 7th USENIX Symposium on Operating System Design and Implementation (OSDI 2004)*, San Fransisco, CA, USA, Dec. 2004. ACM Press.