Adding Privacy in Existing Security Systems

Sam Hartman, Painless Security

When the evolution of the Internet has caused the community to face a new challenge such as privacy, we are faced with a huge legacy problem. Many systems and protocols provide value, but do not adequately address the new consideration. For browser-based applications, evolution to address new concerns is often confined to the organization hosting the application. For example, service providers such as Google and Facebook routinely add privacy-related changes such as new opportunities for their users to customize what happens with their private data. However, other applications, including applications that use HTTP as a substrate, traditional client-server applications, and more complex architectures, may require standardization of protocol changes as well as significant implementation work to address new concerns such as privacy.

The effort to consider privacy in application design shows strong parallels with the earlier (but still ongoing) effort to consider security. These efforts are tied together. One of the first challenges in addressing Internet security has been authentication. As part of addressing that challenge, users needed to be identified so that their credentials could be found and so that appropriate authorization decisions could be made. Today, we find ourselves with a variety of security solutions with an integral concept of user identity. These security solutions are fast becoming part of the privacy problem. However, the same approaches used to develop and integrate these security solutions are likely to be instrumental both for providing privacy protection to these security solutions and the broader problem of integrating privacy considerations into existing Internet applications. This paper briefly discusses past efforts at responding to privacy within the GSS-API, Kerberos and SASL ecosystem as well as describing future plans.

In the early 1990's, there was an effort to develop broad technologies that could help secure applications with little or no modifications to these applications. These efforts appear very similar to the efforts to develop privacy enhancing technologies. The most obvious example of this is Ipsec; the original Ipsec architecture talks about user-level and application-level access control provided at the system layer. SSL and TLS are sometimes viewed this way. However, like privacy, providing a good approach to security requires consideration of security in the application design.

Some applications have developed application-specific solutions to their security needs. Other applications have used security frameworks. The GSS-API (RFC 2743) provides authentication, data protection and naming services. GSS-API is often associated with SASL (RFC 4422) which provides a conceptual way to integrate security services into client-server applications. Both of these frameworks provide some flexibility in naming. For example, both support the idea of an anonymous user. Anonymous authentication was often used when some information such as a collection of messages via the IMAP protocol was offered to the general public. At least initially anonymity support might have been as much because there might not be any credential that a general member of the public might have for a given service as for any desire for user privacy.

One of the first privacy concerns to impact security protocols was the desire to protect the identity of the user from parties not involved in the security transaction. Mechanisms such as Kerberos or some uses of TLS send the client's identity and then cryptographically verify it. As organizations began to be concerned about privacy there was a desire to protect this information. TLS and SSH provide a solution by authenticating the service to the user first and establishing a protected channel over which the identity exchange can take place. Similar mechanisms were adapted by the Extensible Authentication Protocol

(EAP, RFC 3748) for access to wireless networks.

Meanwhile, vendors faced the problem that identifiers such as a username don't make a good basis for authorization. RFC 4768 discusses the GSS-API community's struggle to broaden the concept of naming and move away from name-based authorization. Microsoft's Active Directory platform took a big step away from the traditional naming architecture that these frameworks used. Users are associated with a set of security identifiers. However, these identifiers need not be associated with a user; they can identify groups or attributes of a user. For example one identifier might indicate that a user belongs to a particular organization.

In itself, this evolution did not enhance privacy: identifiers associated with specific individuals are present. However the security community had begun to seriously consider privacy with the development of the Security Assertion Markup Language. In SAML, an identity provider can carefully control what information is released about a user. Later, Bmicrosoft took advantage of the WS security framework as part of Information Card in order to provide claims-based identity management for new applications. Within the security community, we began to realize the importance of evolving our existing technologies to address privacy. For example at IETF63's Kerberos meeting I lead a discussion on how Kerberos could shift to take advantage of these technologies.

Since then, we've had significant success defining standard ways of talking about attributes of an identity rather than a single identifier. We've also had experience implementing anonymity in mechanisms such as Kerberos; see draft-ietf-krb-wg-anon. From a privacy standpoint this is exciting, because we both have the facilities to remove the original assumption of the identifier and facilities to talk about aspects of identity and authorization in the limited to the information that the service actually needs.

Now, we are poised to actually combine these two mechanisms and actually experiment with how legacy applications can be adapted to approach privacy. From a standards standpoint, one example of this work is the newly chartered ABFAB working group within the IETF. We are also working to create the Moonshot test bed (http://www.project-moonshot.org/) where we can experiment with federation of existing non-web applications with a strong emphasis on privacy and data protection. We are unlikely to have results by the time of the workshop, but will be in a position to try out new privacy approaches shortly after the workshop. Also, input into how to evolve existing systems will be valuable to the workshop.
In conclusion, the privacy and security communities have a lot to offer each other. The security community has successful experience introducing new mechanisms into existing systems and applications. The security community has the same privacy challenges as every application. By leveraging the existing security mechanisms, we may be able to simplify the exchange of privacy data. Still, actually addressing privacy will require work on an application-by-application basis. Just as there is no short-cut to understanding an application's security requirements, understanding what user information is needed and how it will be used as well as giving the appropriate parties a way to express their preferences requires application-by-application analysis.