What Next after Anonymity?

Ross Anderson and Steven J. Murdoch

University of Cambridge Computer Laboratory

Abstract

In recent years, interest has shifted from protecting the confidentiality of data to protecting the confidentiality of metadata. The police often learn more from traffic analysis than from content interception; thus anonymity becomes more important than confidentiality in the classical sense. Many researchers have been working with systems such as remailers and Tor in order to provide anonymity of various kinds against various threats.

In this note we argue that it's often not enough to protect the confidentiality of metadata; we may have to protect its authenticity, availability and other properties as well.

The call for papers talked about time travel. This immediately brought to mind struggles with backup and recovery systems. It's convenient if you can let users recover their own files, rather than having to call a sysadmin — but how do you protect information assets from a time traveller? One system familiar to us allows users to recover any file or directory in their filespace, which has the effect of overriding any revocation that may have taken place in the meantime. Worse, a recovery system may give the user access to directories that were always closed to him, by combining ACLs on pathnames. And what if the recovery functionality is buried in an application to which he needs access in order to do his job? This is an example of an example of technological development suddenly providing an extra dimension to the access control space of which sysadmins must be aware; but it may also be seen as system designers paying insufficient care to protecting the authenticity of file metadata.

Backups cause further problems. Suppose you're a human-rights activist in Russia. Two of the many problems you face, and which may become pressing when the police raid you and look at your computers, are that they will find out who you've been talking to — and that they will take your computers to stop you doing useful work. To protect against the second threat you need to keep backups. However, if the cops get hold of your backups, they can compare them with the current disk image, and see that unallocated random data have changed. TrueCrypt is much the same (though perhaps slightly better than StegFS as it's actually used by real businesses for 'normal' file encryption). In this case the metadata property we want to protect isn't authenticity or availability but deniability. A purist might argue about whether the content of the random "unallocated" data is really metadata, but the filesize certainly is.

These points have not been entirely unknown to previous writers. For example, the MLS pioneers in the 1980s remarked that although in theory it might seem possible to build a minimal TCB for a multilevel secure system, in practice one ended up with a substantial part of the operating system having to be trusted once one considered things like backup and recovery in detail. Even further back in the history of security, an example of metadata manipulation might be a soldier putting on a false badge of rank to do something he shouldn't.

How does this affect protocols?

Given that the great majority of real-world crypto failures result from poor implementation rather than from weak crypto or (one hopes, after sixteen years of these workshops) protocol design, we suggest that protocol designers should start to think about how we can minimise the risks that result from tampering with key material. Papers on API security have shown how a protocol suite that contains several individual protocols, implemented on a single device, can be attacked by substituting unexpected keys into invocations of particular protocols. They have concluded that strong, consistent typing systems can go a long way to mitigating these risks.

Other attacks we ought to consider are where a dishonest principal uses time travel to swap new keys with old, and perhaps (where CBC is used) to swap parts of keys by swapping blocks.

It's already known that the classified world obsesses much more than the open one about the destruction of old keys; in December, at the Chaos Communication Congress, someone who'd reverse-engineered the Xbox 360 announced that Microsoft had invested serious money in hardware to ensure that the device could not be rolled back to an old version of the hypervisor. Propagating hard revocation is not as easy as it looks; even given a number of trusted devices with one-way counters, how can we ensure that revocations aren't rolled back, perhaps by some quorum of conspirators? Doing all this robustly may be an interesting research problem.

The second theme for the protocol research community may be how "normal" metadata are dealt with. Traditionally, protocol researchers ignored stuff like email headers: only the contents of the curly brackets mattered. But the world moves on. Remailers may scrub (or encrypt) headers, while Torbutton goes to great lengths to close down as many side channels as possible. DKIM - formerly Yahoo's domain key antispam system — signs email headers, in order to verify the ISP of origin. As a result, it's trusted by the Tor bridge autoresponder, which allocates bridge relay nodes to enquirers. If an opponent could spam this, he could exhaust the pool of relay nodes, or get Tor to spam Gmail or Yahoo, causing it to be blacklisted.

A third theme, and perhaps a more traditional one, is the difficulty of telling what's data and what's metadata. Suppose for example a boss sends an employee a PGP-signed message saying "You're fired!" The victim can simply forward this to someone else, as (by default at least) PGP authenticates only the sender, not the recipient. Of course a thoughtful boss can write "Dear Fred, You're fired!" but this is less than optimal as it breaks a level of abstraction. This is a much more common problem than one might think, as a name at one layer in the stack might be an address at the next, and so on. It's also hard to solve in the general case: PGP, for example, is usually applied somewhere short of the mail server. In theory, one might have a policy of always encrypting as well as signing, and there are indeed technical mechanisms one might use to prevent a recipient combining an old signature with a new encryption. But this is not entirely satisfactory because of the growing complexity.

To sum up: the metadata matter, and not just their confidentiality. It might be useful if we had some systematic thinking about this.