Jikzi: A New Framework for Secure Publishing
(Transcript of Discussion)

Ross Anderson*

Computer Laboratory, University of Cambridge

Jikzi was the first ever book published using a movable-type printing press, it was manufactured in a temple in Korea in 1377 and thus predated Gutenberg by a generation.

Now this is work that I’ve done with Jong-Hyeon Lee who’s one of my research students. What’s it about? Well, we were told that we needed a talk that was relevant to the entertainment industry and also had a strong audit content. Jikzi is a system that is suitable for entertainment in that it is a system for publishing, and publishing arbitrary things, anything from the latest rock-n-roll single through to a great big wadge of public key certificates. What we’re trying to do is build a general purpose publishing system, and there is a heavy emphasis on accountability, on being able, if you are a customer, to be pretty well certain that the instance of the book that you now have sitting in your browser is a valid instance of the book, that it is timely, that it has not been tampered with, and so on and so forth.

Now this brings us to an interesting use here of the policy model, which will pick up on some of the things that we were arguing with Stewart on Monday. There are authentication and integrity mechanisms of course, and one of the interesting things that we’re just starting to develop is a new authentication logic which sits in the hierarchy of things somewhere between Kailar and the BAN logic, in that it does strictly more than Kailar and strictly less than BAN, and, unlike many other things, we’ve got a real implementation underway and it’s fully buzzword compliant.

Where does it come from? Well, at the protocols workshop four years ago I introduced the eternity service: a distributed filestore, which uses anonymity, duplication and scattering, in order to scatter a file over the Internet in such a way that it becomes highly resistant to denial of service attacks. And that’s spawned a number of implementations and a mailing list. There was traffic on the mailing list just this morning, people were discussing how you go about managing directories in eternity. Well, we’ve got some ideas on that.

What we were trying to do with eternity was to replicate the politically important feature of book publishing, namely, once you have published the Bible, or whatever, in the local language, and you have 50,000 copies of it out there, then the Archbishop of Canterbury can’t just round them up and put them in the fire and put you in the fire with them, as happened with people who tried to publish the Bible before movable type printing came to the west. Incidentally, Jikzi was also a religious tract; it was a Buddhist religious book.

* Joint work with Jong-Hyeon Lee

© Springer-Verlag Berlin Heidelberg 2000
So, publishing has politically important features, namely that things can’t be unpublished. It’s also got integrity features, in that if you go to a random bookshop and you buy a copy of a book, the chance that this has been maliciously tampered with in some targeted way by your personal opponent is something that you can generally ignore. And we’ve done some work based on this, we’ve published a global trust register, a book containing all the important public keys in the world. That’s now been taken over by MIT Press.

Another thing, that we did with Fabien [Petitcolas] and Vaclav [Matyáš], was to ask the question: how do you go about publishing a medical book on-line? The particular medical book that we’ve done most with is the British National Formulary, which is a book published every six months that lists all the drugs that can be prescribed in Britain together with the approved dosages, side effects, cross effects, and so on. Now if you were pulling this up into your browser at Addenbrookes as you were sitting there writing a prescription for somebody, you need to know that the information is genuine. And you’re perhaps not worried so much about malice as about all the things that go bump in the night with computer systems, because in the context of an on-line book, these can kill people. Wax was a proprietary first attempt at doing this. We started off with a request from Rudolph Hanka, the Head of the Medical Informatics Unit: we’ve got this on-line publishing application, can you put some integrity on it? So he said to Fabien and Šaček, well it looks like this is an interesting two week exercise for you (laughter). Use X.509 plus a hash of what you expect to find there, and with this very simple mechanism you can replicate many of the trust structures that you find arising for other purposes and from other mechanisms in publishing.

So that’s one of the influences. Another influence is Raj Kailar’s logic of accountability. If you were going to devise mechanisms which will check chains of signatures of hashes, then this appears to be the right tool, at least on first inspection. It’s basically like BAN but without the freshness. You’re asking the question whether somebody ever signed something, rather than whether he’d signed something during the current protocol run.

Another thing that has given some input to this is Bruce Christianson and Bean Snook’s Distributed Object-based Document Architecture which was also described here about six years ago. This is a system that’s a bit like a secure RCS, in which you have a number of elementary documents, which are chapters of a book or pages of source code or whatever, and these are held together by folios, which are structures that contain the necessary pointers or makefiles or whatever. And these are integrity protected using hashes and signatures, and there are various demons which can automatically execute certain publicly agreed security policies – such as: a version may be updated if one user manager and one software engineering manager both agree – so you can have automatic demons which will see to it that an update occurs as soon as the relevant signatures appear somewhere in the universe.

And finally, of course, we have to be buzzword compliant, and so you’re supposed to do things like adding new security tags using XML (extensible