CS 240: Cryptographic Hashing Applications Transcript

[00:00:00] Next, we'd like to talk about a few applications of cryptographic caching that, that um you'll find in real world applications. So, um on this slide, we see a couple of them. The first one is that we've been using GIT as our um version control system in this class. And as you're probably aware by now, every time you do a commit and GIT there's an id that is assigned to the commit. So, every commit has a big, long string that looks random and that's the idea of the commit.

Start visual description. The professor demonstrates the use of GIT as a version control system. On the screen, the professor shows a commit in GIT with a unique identifier generated by the SHA-1 algorithm. The commit ID is a long string that appears random, representing the changes made in that particular commit and the state of the source tree at that point. End visual description.

[00:00:33] So if we ever need to refer back to that commit, we, we have a unique, unique identifier for it.

[00:00:38] And um it's interesting to know that the SHA one algorithm, the hash algorithm is used by git to, to generate those commit I DS. And so that's just kind of interesting to note, but that's an example where it doesn't have to be super secure.

[00:00:55] I mean, these just have to be unique I DS.

[00:00:58] Um But a hash algorithm is, is a pretty good way to generate a unique id um based on the contents of, of some data.

[00:01:05] So in this case, the data we're talking about would be the, the changes that were made in that particular commit and the state of the, the source tree at that point.

[00:01:15] And so they, they generate a hash value from that.

[00:01:19] A, a second application is verifying the integrity of files that you might download from the web.

Start visual description. The professor demonstrates verifying the integrity of downloaded files using SHA-256. On the screen, the professor shows a screenshot from Oracle's Java Development Kit download site. The professor clicks on the SHA-256 link next to a file to display its hash value. The professor explains how to compare this hash value with the one generated after downloading the file to ensure it hasn't been modified. End visual description.

[00:01:24] So a lot of times we'll go to a website and download a file of some sort, maybe it's an installer for an application, maybe it's a data file or whatever it might be.

[00:01:34] And especially if that file contains executable code, you might want to try to verify that that file has not been modified in transit that it hasn't been tweaked some somehow either intentionally or randomly and that it, it's in its original form.

[00:01:52] And so sometimes you'll see websites like this, this example here on the slide, this is uh a screenshot from the oracle's Java development kit download site.

[00:02:02] And you can see here that you can click on uh various links to get different versions of the JDK.

[00:02:08] But you'll notice out to the right um at the end of each line, there's a little link called SHASHA 256.

[00:02:14] And so if you want to see what the, the hash or the message digest is for that file, you can click on that link and see what the SHA 256 digest of that file is. So that um once you download the file, you could, you could run that file through SHASHA 256 yourself and get the message digest yourself and then you could compare the two and make sure that they're the same because uh you know, if they're the same, that means the data hasn't, hasn't been changed.

[00:02:41] And so that's another uh kind of security application for hashing algorithms.

[00:02:48] Um In general, we deal with data a lot obviously in programming.

[00:02:53] And so any time you need to create a compact unique summary or identifier of a block of data, uh these hashing algorithms are a great way to do that. And in the same spirit as GIT did with their commit I DS. And so, um for any number of applications, it's just useful to be able to take a really big piece of data and then have a small fingerprint that, that represents that data that is pretty unique and pretty unlikely to um be duplicated.

[00:03:27] Um Secondly, anytime you want to make sure that data hasn't been changed in or you know, in transit across the network or even at rest in a database, either way if you want to have a way to verify that the data hasn't been changed, you can use these hashing algorithms for that.

[00:03:44] Um It might be interesting to know that Bitcoin, the Bitcoin Mining algorithm uses SHA 256 as part of its algorithm.

Start visual description. The professor demonstrates the use of SHA-256 in Bitcoin mining. On the screen, the professor shows the process of generating new Bitcoins by producing a piece of data with a specific message digest format. The professor explains how the computer runs the Bitcoin mining algorithm to

generate values that match the required message digest format, highlighting the computational power and energy required for this process. End visual description.

[00:03:52] And so the Bitcoin Mining algorithm is, is an algorithm where you want to generate new bitcoins. And that's how we can produce uh new Bitcoins is, is we have to um essentially the goal of the mining algorithm is to produce a, a piece of data that has AAA message digest that has a certain format that has certain digits that have certain values. And so, so what your computer would do when it runs the, the Bitcoin Mining algorithm is it would just sit there and generate things until it happens to generate a value that has a message digest. That looks the way that the algorithm wants it to look. And that's how you can generate new Bitcoins.

[00:04:33] But it's not easy to do. It takes uh compute power, it takes energy.

[00:04:39] And so it's not too easy to generate a Bitcoin, which is important.

[00:04:42] Otherwise we'd have way too many of them.

[00:04:45] Um Another place hash functions get used is in secure user password storage and we'll talk about that uh in detail next.

[00:04:53] But we want to make sure that we securely store people's passwords so that they can't be compromised if somebody breaks into the system, um they can't figure out what people's passwords are.

[00:05:02] And so hashes are, are good for that.

[00:05:05] And also hashing algorithms are core to the, the way that digital signatures work.

[00:05:10] So in a digital environment, we, we sign documents that are digital.

[00:05:15] And so we have to have an ability for somebody to sign a document in a way that can't be repudiated. And so hashing algorithms are also useful for implementing digital signatures, which we will also talk about later.