



*Announcer You're listening to the Sun Microsystems Podcast Network.*

*Announcer You're listening to the Sun Microsystems Podcast Network. Welcome to another edition of Innovating@Sun with your host, Hal Stern. Today's topic: ECC interoperability. And now here's Hal Stern.*

**Hal Stern:**

**Hello and welcome to Innovating@Sun. I'm your host Hal Stern, Vice President of Global Systems Engineering at Sun Microsystems. My guest today is Vipul Gupta, a distinguished engineer in Sun Labs, and welcome to math class. Vipul, you are leading an effort here on interoperability of Elliptic Curve Cryptography, so I think the place to start really is with the math lesson - tell us a little bit about what ECC is.**

Vipul Gupta:

Maybe the way to get started is to say what ECC is not because a lot of people hear the word ECC and they think of error-correcting codes, so when we talk about ECC we're talking about something called, Elliptic Curve Cryptography and at a very high level this is a new kind of cryptographic technology which is stronger and much more efficient than the more commonly used counterpart which is RSA.

**Hal:**

**I think that when people think of cryptography they think of very large exponentiation, modular arithmetic, things that that basically consume an awful lot of cycles on your typical processor. What's different about ECC? What's the mathematical foundations?**

Vipul:

Right, so with any kind of cryptographic technology, the strength of the technology depends on how hard it is to solve the underlying math problem, and for RSA, that math problem is factorization of large integers. ECC relies on a different kind of math problem which is called the elliptic curve discrete logarithm problem. We don't have to go into the details but the point is that the best known algorithms for solving the ECC problem run much more slowly than the best known algorithms for solving the integer factorization problem. What that means is that ECC can now provide the same level of security as RSA using much smaller keys. When you have smaller keys you can get faster computations and you also save up on resources like memory, bandwidth and energy.

**Hal:**

**This really comes down to preventing brute force attacks. If it's easy for me to go run an attack and try to strong arm my way into reverse engineering the keys if I know the algorithms, the harder the algorithm is the slower it runs, clearly, the more secure that that result is.**

Vipul:

Exactly. So you could use that either to use smaller keys to provide equivalent security or you could use that to gain better efficiency in terms of how fast your computations run.

**Hal:**

**Just as a bit of a baseline, today we think of the RSA algorithm using 1024-bit keys and in the next couple of years, just to stay ahead of the state of the art in brute force grid-type attacks. The key size there is expected to double, so you're going to be looking at least 2000-bit keys by the year 2010. What's the equivalent key length you would need to use with an ECC implementation?**

Vipul:

For the 1024-bit key size that RSA currently uses, the equivalent strength is provided by a 160-bit ECC key. And for the 2048-bit RSA keys that NSA is recommending people move to by 2010 the same amount of strength could be obtained by using an ECC key that's only 256 bits long.

**Hal:**

**So not only is it stronger with an order of magnitude smaller key but the key length is not increasing at the same rate as a RSA key would to get the same level of security?**

Vipul:

Exactly. And if we plot how ECC key sizes need to grow compared to RSA keys sizes to provide the same level of security, then we see that ECC grows a lot more slowly than RSA. What that means is as your security needs increase even beyond 2010, ECC starts looking more and more attractive compared to RSA. That's just the key size thing, the more important thing to keep in mind is that performance is actually cubic in the key size. So when ECC key sizes stay below RSA, the performance gain that you get by using ECC compared to RSA actually increases at a much faster rate.

**Hal:**

**What I'm thinking is that as Sun talks about more and more devices and people being connected to the network, clearly we want many of those connections to be secure. I'm going to be limited by things like compute power and battery power and mobile battery lifetime on whatever device I have, the less I'm spending on computing my session keys, probably the happier I am.**

Vipul:

Exactly. You see one of the things that ECC is best for is these new kinds of computing devices that people call wireless sensor devices. If you look at the evolution of computing over the years from mainframes to desktops to laptops and then more recently phones and PDAs, we see that with each generation of new computing devices there's a much larger number of those coming on to the network. And we feel that the next wave of these computing devices will be things like the Sun SPOTs, which you might of heard of. These are matchbox-sized battery-powered wireless devices being developed at Sun Labs. And there's other companies and research groups that are doing similar things. With these kinds of devices many of them have as little as four kilobytes of memory. We've played with some of these and we found that even with current RSA key sizes which is only 1,024 bits, you can't even put RSA computations on these devices, let alone leave any room for applications on top of that crypto computation. But with ECC we were able to actually put not just the ECC computation but an SSL stack and some additional applications on top of a coin-sized device called the mote, and that resulted in the development of the world's smallest secure web server. So for these kinds of new devices, ECC is just phenomenal. In many cases it's the only way to go.

**Hal:**

**But I think the fact that it could be pervasive raises the other part of the problem which is, here you have a mathematical standard that has to get reduced to code. How do you make sure that one implementation of ECC**

**talks to another one?**

Vipul:

Right. You bring up a very good point. With technologies like this, the real value comes from having this not just on one or two devices but on as many devices out there. You're all familiar with the concept of the network effects. The value of any technology grows as the square of the number of devices that implement it. This is something that's been on our mind for a long time so when we started looking at ECC as part of the Next Generation Crypto project at Sun Labs, one of our goals was always to make sure that ECC is widely deployed. And as part of doing that we've done several things. We have not only incorporated ECC into Sun products, we have made contributions of ECC to open source, so both OpenSSL and the cryptographic library that's used by Firefox and all of the Mozilla products have ECC functionality which was donated by Sun. That was one aspect of it. We have also been very active in the standards organizations, primarily at the IETF. We looked at SSL, which was the dominant security protocol on the network today and what we did was we added ECC functionality to it. This resulted in an open standard, and because the standard was open it's now being implemented by companies like Microsoft, Red Hat and a whole bunch of others.

**Hal:**

**So what's the ECC Interoperability Forum? Is that an umbrella organization to make sure that everybody plays nicely together?**

Vipul:

Actually what's interesting about this ECC Interoperability Forum is that it's very free-form. There is very little administrative overhead. It started off as just three folks, myself, Bob Lord from Red Hat and another person (Ari Medvinsky) from Microsoft. We were just talking about things and we all knew we were implementing the same protocols and we wanted to make sure that we wouldn't have a repeat of something that's happened in the past. For example, when people were trying to migrate from SSL Version 3.0 to something that's a little more secure and more standardized called TLS, there were interoperability issues between Microsoft's implementations and the implementation that Mozilla has. So as a result, what happens now is even though both sides claim to have TLS in them, when you actually try to connect from a Mozilla browser or a Firefox browser to a Microsoft server the two are not able to negotiate TLS and then they fall back to SSL 3.0. As a result what's happening is that even though there's effort put in by both sides to implement TLS in their products, customers are not actually getting the benefit of that effort. So with ECC we wanted to make sure that when ECC gets into those products, we wanted to resolve all the interoperability issues before the technology gets widely deployed, so customers can start taking advantage of the benefits that ECC brings almost immediately.

**Hal:**

**Great! I think as you put it, this effort goes beyond just the major commercial software players, various government agencies are now looking at ECC as a way of protecting public information effectively.**

Vipul:

Right. The main government body that kind of dictates where the security world is heading is the National Security Agency, the NSA. In the past some of the things that NSA has endorsed have gone on to become widely used standards. For example, Data Encryption Standard (DES), was developed in the mid '70s and in about early 2000 they realized that the keys that DES was using were no longer big enough to protect the information in today's world. So they proposed the use of the Advanced Encryption Standard or AES. What's happening now is something similar for public-key technology. Both DES and AES are examples of what's called symmetric-key technology which means that the two communicating parties use the same key to do their communication. Public-

key technology, which is something that you use in conjunction with symmetric key technology, is used to actually establish the symmetric keys on two sides across an insecure network. What the NSA has done now is come up with something called the Suite B. Suite B dictates ECC as the only public-key mechanism, so they've completely dropped RSA, because they feel that the kinds of key sizes you would need to provide adequate security in the very near future would just make the RSA performance too slow.

**Hal:**

**Give us a little bit of history lesson here, in terms of the Next Generation Cryptography team within Sun Labs, this is not something that you started on recently, this goes back a way in terms of thinking about where we want to go with ECC and how it was going to intersect our product line.**

Vipul:

Right, this goes back to almost five years or so. In the next generation cryptographic team we had four people, Sheueling Chang, Hans Eberle, Nils Gura and myself. We all came from different but very complementary backgrounds. Prior to coming onto the next generation crypto project I was working on trying to put standard Internet security protocols into phones and PDAs because back then the common wisdom was that Internet security protocols were too big. So organizations like the WAP Forum which were promoting something called WTLS, which is completely incompatible with SSL, and as a result they did not have end-to-end security when people were trying to communicate from mobile devices. The project I was working on back then put a tiny implementation of SSL, which was the first one small enough to fit, into those kinds of devices, but that was all RSA-based. I was thinking, that works for the client side because RSA has this asymmetric nature, but if you want to have these devices be accessible from the outside, and act as servers, not just the clients, then RSA isn't very good. So I was looking at ECC from that angle. Sheueling came to ECC from the viewpoint of, it was an interesting math problem and she's a brilliant mathematician; and Hans and Nils came to it from the angle of, how can we speed up certain mathematical computations. Hans, in the past, has done hardware that speeds up DES encryption, for example. So we had this team - I like to think of this as a dream team - as we sometimes joked, because we had this fortunate coincidence that there were all these people with very complementary backgrounds and they all had, all of a sudden, this common interest in this new kind of cryptographic technology. We started looking at what kinds of architectures we could build around ECC, not only for Sun servers, which you would tend to think of as very different from wireless sensor devices. But many of us were also interested in and could already see these new kinds of devices that would appear on the horizon in the short future. We could also see that when you have so many of these new kinds of devices coming on, you would automatically have new kinds of applications. For example, before a lot of computers got on the Internet, applications like banking were not online. You had to personally go into a bank and conduct your transactions. So when new kinds of devices and more people get on the network, new kinds of applications migrate to the Internet, and many of these applications have security requirements. When you combine new kinds of applications requiring security, more transactions requiring security, and you also have devices that are not as capable in the sense that they're simpler, then you have an interesting problem. You need something that is very efficient and this is where ECC fits right in.

**Hal:**

**So you went in two different directions. One is that as we drive more and more content onto a network, there are things that we've always thought of as data sources, that now we may want to think of as servers. That we're actually going to go to them and poll them and collect data from them, and they're actually going to be not just sources of data, but actually doing something with the data. And you've given the example of the Sun SPOT, or the embedded web server, so clearly, we need better crypto there. The other one is where does this intersect our product line? So can you tell us a little about how ECC can make it down - basically down to the silicon level?**

Vipul:

Sure, yes. On the first of those points, it's definitely true that these kinds of devices would not only be clients but also servers. In the overall architecture, we think of information coming back from these devices. That's all kinds of sensor information, but we also see control information going to these devices. So the information flow is both ways and either side could be initiating that connection. In that sense either side could be acting as a client or a server. That's why it's no longer enough to hide behind the asymmetric nature of RSA and use RSA. ECC becomes extremely important in that situation. In terms of ECC and Sun products, again we are trying to approach it from two sides. Now that you have this more efficient cryptographic algorithm, you could use it to support a lot more transactions on a device that has certain capabilities. What we're doing is putting ECC on in the Sun Java Web Server. So 7.0 came out and that has ECC functionality; as a result now, you can have an ECC-based web server that can handle many SSL or TLS transactions compared to a web server that only implements RSA. We also have ECC now implemented in Java SE 6.0 and on the small client side we have projects in the Labs that are using ECC in interesting ways with the Sun SPOT. For these kinds of devices, many of the applications are unmanaged applications in the sense that there is no dedicated system administrator for these devices. Think of applications like monitoring redwood trees, or monitoring a battlefield. In these kinds of applications these devices are often placed in places that are not easily accessible. And if you have a bug in the code or if you learned something about the environment and you want to change the software on these devices, what you want to be able to do is reprogram them over the air, because you won't have too many people who want to volunteer to go out in a battlefield, pick up all these devices just because there's a bug, bring them back to the lab and reprogram. So you want reprogramming over the air, and when you want to do that, of course there are security concerns. In the past, people have felt that these kinds of devices could only do symmetric-key cryptography. The problem there is that now you have the same key that's on the device as well as on the controller, which is in a safe place, but these devices are small and can be easily compromised. If you're using the same key both ways, you compromise one of these devices and now you can pretend to be the controller for the rest of the network. With asymmetric-key cryptography, you only have the public key of the controller on these small devices and the controller is issuing commands or gathering data and it's all signed using asymmetric-key cryptography, so it gives you a lot more security guarantees when you have asymmetric-key cryptography for these kinds of applications.

**Hal:**

**And as with most things, there's not a single answer here, and we're gonna use a mix of asymmetric keys to go establish a session key. We may use a session key to go do bulk communication, but we have that flexibility now and we have that flexibility without trading off strength or ease of implementation or anything. In some case we have the possibility of implementation given the work you've done in driving down the footprint, the compute footprint, the memory footprint, the time footprint of the ECC algorithms.**

Vipul:

Absolutely, I mean in any security protocol whether it's TLS or IPsec or SSH, you always use a combination of public-key cryptography and symmetric-key cryptography. Public-key cryptography is typically the part that takes the most resources. Now with ECC, you could reduce the kinds of resources you devote to public-key cryptography. That just opens up a lot of new applications.

**Hal:**

**So I guess I have to ask you, I think this is certainly a new field for a variety of practitioners in the security space, why isn't this as well studied as let's say the Diffie-Hellman key exchange or some of the RSA protocols, the other things that rely on the module arithmetic and on exponentiation? What was the major event that brought ECC into the forefront?**

Vipul:

ECC is actually not that new, which is a good thing because when you're dealing with security protocols, old is gold and you want security protocols that have been around for a while so that people have had a chance to look at them, kick the tires and try to poke holes in them. ECC was actually invented in the mid '80s, in about 1985, by two folks independently. What makes ECC different from other algorithms that we are more used to like RSA and Diffie-Hellman, is that unlike doing computations that involve large integers, it is doing computations on elliptic curves. You have the notion of points on a curve, you have the mathematical notion of adding points and then when you do repeated additions you can do point multiplication. So that's what makes ECC different, I don't want to say it's new, but it's newer, but it has gone through a lot of reviews and people, including the NSA, feel quite confident about deploying ECC as a replacement for RSA.

**Hal:**

**About a year ago Bernard Chazelle, a Professor of Computer Science at Princeton, was lamenting the lack of interest in computer science versus the growing interest in mathematics. And here what you've talked about is, instead, mathematical tools that enable a whole new set of computer algorithms, what's next on the algorithmic side? Where do you think you're gonna take ECC next?**

Vipul:

In terms of what we're trying to do at Sun Labs, we are looking at applying ECC to the new kinds of devices, primarily the Sun SPOT; we already have some ECC functionality built into Sun SPOTs, primarily around code dissemination over the air. And we're now looking at how we could extend that to secure the communication between these devices, so that's what we're looking at in the immediate future on the small device side. On the product side, something that's interesting that's coming up is the inclusion of ECC functionality into the Niagara 2 processors, so the product teams are trying to make sure that our software applications can take full advantage of the ECC speedup that Niagara 2 brings with it.

**Hal:**

**Great, so if you had to come down right now as a mathematics guy or a computer science guy, more interesting on the computer science side?**

Vipul:

Definitely, yeah I think math is just a tool and I actually personally like to see things that people interact with in a more direct fashion, so I am definitely on the computer science side of things.

**Hal:**

**Okay Vipul, thanks for giving us this view of the future of cryptography and embedded devices that understand where that future is going. You've been listening to Innovating@Sun, and I'm your host, Hal Stern.**

Vipul:

Thanks, Hal.

*Announcer      You've been listening to Innovating@Sun. Join us next time for the latest in innovation from Sun Microsystems. Only on the Sun Microsystems Podcast Network.*