Farinaz Koushanfar: The fact is that when we talk about protecting patients, we're not talking about every privacy problem that I have worked on during my career. There is a big complexity when it comes to health. It definitely concerns every human, so everybody is concerned. It is safety-critical. We don't want to make mistakes. It's fragmented. The data is inaccessible. There are a lot of compliance issues, privacy issues, and so on, and security aspects. We want to make sure that the data that we have, the models that we have are making predictions, making assumptions, the devices we have are not attacked. Many of these problems are at the forefront of what I have been doing research on during the years, and there's definitely a competition among different actors in this space. The state of the affairs today, when you talk to many healthcare professionals and many hospitals, and I've tried to do this for a very long time, the fact is that you come to the point that healthcare data is not just any other data. It's biologically unique. There is a balance between the utility and how much you're exposing the human data. It is not a commodity, even though a lot of entities are trying to commercialize it. There is a bias strap, I think Aaron Deep really articulated it well during his talk when he talked about the marginal populations, how they actually get marginalized in these type of scenarios and models. There's a fragility of trust. We have on the one side, the healthcare system, and on the other side, the users, and then the trust between these two entities, even given the compliance is not really clear model. The exposures are irreversible. It's not just like your credit card got exposed and you can change your credit card in one day. Those things make things much more complex than they are, and there are barriers to building ethical AI today. We have the silo problem. There are different entities. Each of them have their own data there is the proprietary wall. We can't really access it. There is a regulatory labyrinth. I sit on several standards for these technologies, and it is just so hard to navigate, even for people who are establishing these standards. There's a re-identification risk for the private data, and added to that are privacy attacks. A lot of people have interest in patient data for various reasons, and that creates a lot of possibilities for attacks. Then, on the top of it, the data that comes from different healthcare providers and different health entities are not compatible. They're really having this issue. Even if you are willing to work together, this issue incompatibility is making things rather difficult. When you talk about confidentiality today, a lot of times what people refer to are what is known as a algorithm called secret key cryptography. Basically, Alice and Bob both know a secret key, the same secret key. What Alice does, Alice is going to encrypt whatever they are trying to encrypt with that key, and then they're going to send it to Bob, and Bob is going to just unlock it with the same key, and they're done. Added to that are the protocols that are known as public key cryptography. They improve the state of the affairs. But still, what remains as a problem is these protocols are mainly targeted to secure communications. There are other parties that get involved to ensure confidentiality. There is a concept known as a trusted third party, when you don't really directly communicate to someone with your private data, but there's a trusted entity that does that. Think Experia. You don't share your credit history with everyone, but they contact that company to do this, and it has its own issues, as you have seen in the breaches and other things that has been seen. This is also a model that is really fragile. What is really missing in encryption today is we know how to encrypt communication. We know how to encrypt storage. When you store data, you know how to encrypt the stored data. But what is really missing is the computing piece of the puzzle. When you're computing, you always need to compute on the data that you see. It is in what we call a plain text. It's not cipher text. You cannot compute on a cipher texted today. That is a big issue. I guess the great news is that there are protocols called overarchingly secure two-party and multiparty computation that work exactly like a black box magic. Here we have two parties, Alice and Bob. Alice has an encrypted version of its data. Bob has an encrypted version of its data. They jointly compute on this encrypted version, and neither of them ever gets exposed to the other party's data. This sounds amazingly fantastic. The multiparty version is just an extension of this. You see it up there. There are multiple parties that are getting together, and they can compute on private data. There's different adversary models that you could you could imagine with this. There is the passive adversary, where the corrupted party can just eavesdrop on the communication, try to figure out what's going on. There is the active adversary

where you just are one of the parties. You could actively deviate from this protocol and then try to cheat and get more information. Then there is the active adversary, which is covered. The detection of the misbehavior has to be with a consistent probability in this model. But I guess this is too much technical details. What I want to talk about now is the fact that we have the theoretical results to do this today. There are these protocols called cryptographically secure multiparty protocols, and I'm just giving a very selected timeline of the amazing events that has happened in this field. Some of the brightest and most brilliant minds of our times have worked on this problem. In computing, there is something called Turing Award, which is equivalent to Nobel Prize in computing. What I'm talk about is our people who have raised the foundation for this field and have got this type of prizes. I'm not going to since probably the audience are not very familiar with the field, I'm just going to go through this, but, rather quickly, one of the biggest breakthroughs in computer science Adleman came in 1976, when the public cryptography was almost simultaneously introduced by Diffie and Hellman and Rivest, Samir, Adleman, and after that they all of these people have got Turing Awards, so I'm not going to go through that every time. Then there is privacy by homomorphism that came on, then secret sharing of libs transfer, then Yao's garbled circuit, zero knowledge proof. I guess, you know, the only person who hasn't got it during award is Craig Gentry, who invented this as a PhD thesis in 2009, so he still has time to receive it. But what I wanted to say is that we're really living on the shoulders of giants. There are amazing mass, amazing cryptography, amazing results for how this can be done. Now the question is why it hasn't been done or not fully done. I guess public key cryptography is used every day. You all use it every day. These are used for communication security, but what I talked about computation security is the part that is missing. I'm not going to go through the details of this. This is too much details technically, but I wanted to talk about is probably privacy preserving AI and how these protocols have really transformed themselves in recent years from really being theoretical to something that has become much more practicable. I'm going to specifically talk about one of the protocols I discussed, which is used for both two-party and multiparty computation. It's called the Yao's garbled circuit protocol, for which, of course, Yao got a Turing Award. But here, I'm just showing a selective timeline. The protocol itself was introduced in 1982, but there was no way to practically implement it until 2004. Even then, it was really, really inefficient. Since then, we have seen a number of realistic compilers that have come to try to solve the problem. But there is a big problem with this protocols, and that is they're computing directly on bits on zeros and ones. When you have a complex function, computing on bits, as you could imagine, takes a very long time. I guess it was just a third and DPT that I started looking at this problem with my hat as a computer engineer. Of course, as you know, as computer engineers, on a chip this small, we were able to do tera of operations. That is because we know how to synthesize Boolean logic very efficiently into very compact spaces. I guess I got very lucky because everybody else who has looked into this problem was either a software engineer or a cryptographer. They've never really thought about, Oh, wow, this problem can be solved. This was the birth of TinyGarble, which is a project I initialized and worked with, and then a series of practical results came in the field where you could do this Boolean optimization very efficiently. This was basically done by mapping to a hardware synthesis technique and it brought in scalability and orders of magnitude compact presentation to the field. It could generalize to other private circuit-based methods, and we and others have built on that and enabled a lot of applications that have never been enabled before, including genomic matching, secure locationing just matching of images. If you're familiar with the US residency match, that's a stable matching algorithm. How could you compute on private data without ever sharing and without having a private party that does this type of matching? Several of these applications were enabled for the very first time after we introduced the scalable method. I'm not going to go through all the details of this because it's probably not as much of interest to the audience. We have a number of papers throughout the years on these topics. But the one that I wanted to specifically focus on is a problem that has been of big interest, and that is deep learning on private data. Here, we have a user or a client and a cloud server, and the user wants to compute its data on the model that is available on the Cloud, but never wants to share the data with the Cloud. Here is an example with Microsoft as your Cloud, but you

could make that example with anywhere you want, and then you basically get your black box magic. And again, after the introduction of TinyGarble, this was one of the very first problems that I worked on, and I guess the logic was, we can now garble any computation. Why not computations in deep learning? Basically, you just lift all the computations that happens there, and you go, and you find a Boolean expression one by one, which was amazing because for the very first time we enabled this application. But again, as you could imagine, we are talking about networks with hundreds of thousands of parameters. If you were just mapping to zeros and ones, good luck to have a practical implementation very fast, very soon. Then the challenge was also the data, when you encrypted it, it became much bigger. There was basically delays that were associated with this because of all the computations that was going on. The way to try to go around it is by making algorithm co-design. We modified parts of the algorithm and parts of the protocol to securely execute the functionality. Again, I'm not going to go through all the boring details and bore you with the boring details, but we have been able to scale this with a hybrid of multiple protocols. Some of these works are my work, particularly the two last ones. For the first time, we were able to make the round complexity constant while we were able to have resiliency to active security. I guess these are two technical terms. I'm not going to go through all the details of this. What is really important is that we can scale things when things are not linear with the computations, because you can precompute many things and then have the constant round help you make things become very practical. I'm going to switch gear a little bit, talk about another protocol that is super important. Another Turing Award, of course, that has the potential to again revolutionize computing, and that is the zero-knowledge proof protocol. This is something that I have been working in the past three, four years. Again, another black box magic here, we have a prover which has done some computation but doesn't want to reveal the results of the computation, but they want to prove that they have done it. Imagine solving a very hard problem. You have the solution to the hard problem. You don't want to share that solution with anyone, but you want to prove that you have the solution to the problem. You could use this for verifiable computation. For example, when somebody tells you, I've done this computation for you, but they haven't done it, you could catch it because that's what this algorithm does. Basically, here, the weight lifter is lifting the weight. They're not going to share how much weight they lifted, but they can prove that they've lifted this weight with you. Here's another example, where is Waldo? In this picture, you're looking for a person called Waldo. Here, naively, if you want to do this, you just circle his picture, and then you don't show it the other parts to the others. You just say, Waldo, this is him in the picture. But obviously, this is a very naive solution. In cryptographic world, this happens in a much more sophisticated way, just to give you an idea of how this type of algorithm works. Again, when I started looking to this problem, there was no practical framework for this, and we were able to take this from theory to reality this time in collaboration in my role as a scientist at Chainlink Labs, where we had the very first practical implementation of this in real-world. But we've done a lot of research in MyLab before and after this problem. There's a surge of ZKP-based applications in recent years. I just told you about the blockchain applications. There is a verifiable computation when you do inference and just learning. Last but not least, there is a lot of work on, as you know, federated learning that I will be talking about shortly, and I'll talk about how ZKPs or zero-knowledge proofs can help with making them both robust and private. In many scenarios today, we have centralized systems. When you have a centralized system, there's a single point of failure, and there is a mass breach of secure data. There is a limited user control. There are different attacks because one person has a lot of control over this whole thing. There is a slow response to threats, and there is censorship, there is manipulation by big entities and so on and so forth. I was fortunate to be part of the founding team for the federated learning effort, which which was basically led by a number of researchers at Google, but a number of academics, including me, have contributed to the landmark work and results that are now being used everywhere. Here, in federated learning, you're trying to do collaboration without sharing the data. Basically, each of the clients are learning their own model and then they're aggregating the model parameters too to figure out what's happening to update one model and the model gets updated this way in a distributed way, you're collaborating, but you're not really sharing

your own data. There are different security aspects to this federated learning. You may have one of these clients become a bad actor. They may just disturb learning by poisoning the scenario, they may try to have a targeted attack when they intentionally provide values that completely ruins everybody else's computation and so on, so forth. One of the main results that has really helped the federated learning gets a lot of traction was these methods based on that are called privacy preserving federated learning, where they try to do aggregation, which is basically a summation in a private way. You can sum up values and use the sum without knowing the individual components, but you really don't get to know the individual components. This is great because it works really well. It's a simple algorithm. It can scale very well. It has been beautiful. What was missing in the original algorithm and that's where algorithms such as zero-knowledge proof that provide verifiable computation help with is the fact that if somebody is lying or somebody is providing a poisonous update, if you're doing all the computation in private, you would never be able to catch them. That becomes a huge robustness issue. That problem is really central to safety of these type of federated learning scenarios. It's not just targeted poisoning or poisoning, but also outliers. If you have outliers and you're building a federated learning model, you are very prone to this type of scenarios. Attacks during local training, attacks could happen during just the data reconstructions and redistribution and so on and so forth. Here's some work we did with privacy preserving federated learning, which is resilient to Byzantine attacks where you have this type of deviations in sets of data. We were able to address this problem in malicious threat model by incorporating zero-knowledge proof. Basically, you don't see the value from people, but you get a proof that this value was robust and within the range that you are looking for. That's the overarching idea. The details of the algorithm will be boring to you, but it scales sub-linearly with respect to the number of clients, and it's able to remove outliers. Basically, again, a little bit too technical of a term, but rank-based statistics are one of the main methods used to overcome outliers in scenarios. Now, this was a bit more simplistic than what we're really looking for intending in future systems. We're talking about federated learning, and we will not just have people. We will have agents. We won't have agents that are as was mentioned are able to do health for us, like the previous health decisions or health consultation for us. You want to train these without sharing data, but you want to collaborate among these actors. Things are not as static as it was in the federated learning scenario I originally shown. This dynamism makes a lot of things super challenging. Inference also becomes a bit more challenging. What is really important in these systems we are agentic based, then you could have these automated agents that are able to collaborate, but don't share the data is you could actually multiply, or you could have heard for their experts, and then you could basically multiply yourself to ensure scalability. The challenges in distributed collaborative learning systems are not just the ones I mentioned. There are a lot of challenges with secure data transmission, adversarial attacks, model inversion, transparency, access control, regularity, compliance, and others. I'm not going to have time or I don't actually work on all these problems myself, but if you're interested to discuss, I'm happy to talk about. Really, what I want to say is that secure distributed collaborative AI is here and it's going to stay. We have several technologies that can help with making this. Technology I didn't talk about today is trusted computing or secure enclaves, which has its own advantages and disadvantages. The things I talked about were zero-knowledge proof, secure multiparty computation, and I didn't talk about homomorphic, but hopefully in another talk, I can if I ever get invited again. Another type of learning that we have looked at is split learning here. It's a little scenario is a little different than, and then the federated learning in that federated learning, the whole model gets updated by distributed clients. Here, you basically split the network or split the learning parts of the network to different clients, and it works actually well when you have non-IID data. It works better than federated learning in terms of convergence, and then you try to calculate loss and then do a distributed back propagation to try to have a convergence and have the forward client model. Again, here is a threat model where you could have untargeted poisoning and targeted poisoning attacks, similar to the federated learning approach. The problem here in federated learning, we had some a check from other learners for comparison here. We don't have that because different parts are learning different things, and this

was, again, a huge challenge. We looked into another zero-knowledge protocol to solve this problem for split learning. Recently had a paper at the CCS 2025, which is a top conference in security privacy. If you're interested in taking a look at that and then we did a proof of concept in a huge data server and on edge devices in our lab to make sure that this is practicable. Again, last but not least, I wanted to just talk about or brush upon a very major challenge, and that is the data. Basically, we're running out of data. As you know, a lot of data that we have now is synthetically generated and the problem is not improving, but this is mostly about the surface data. 95% of the data actually is what is called a deep web, and deep web are data that is just password protected. Your data in the hospital, health records, emails, financial records, things that you can never access or at least visibly access and try to get a control of. The question is, can we use this data without ever exposing it? Can we use it for inference? Can we use it for learning? I don't have time to talk about this, but we have an approach called props for machine-learning in collaboration with my collaborator Ari Juels at Cornell Tech, he used to be the head of RSA labs. This is, I think, a really promising method. Let me just wrap up. There are several ongoing research and directions that we're taking things from here. I briefly talked about autonomy and autonomous agents. You could also use these autonomous agents to automate security, streamlining zero-knowledge proof, secure multiparty computation and other things. This is some of the work that we're doing now. You could use these blockchain-based methods that have some of these protocols already implemented to ensure data provenance and protect the pipelines. As I said, you could use it to train on deep web data and to infer on deep web data using these type of protocols. We're also looking in clinical robustness and AI's integrity. There are high stake medical environments that could be subject to different types of attacks and trying to have safeguards for that. We have a number of recent collaborations and health initiatives within UCSD, Health Sciences. Some of the problems we are working on are focused on federated learning when you have data from different hospitals across, I guess, not just UCSD and not just UC system. We're even trying to go outside that and try to do that. Then edge security when you have embedded devices, when you have patients that are relying on those type of devices without before even going all the way on the cloud, how do we address some of those problems? I know I went too long. I just want to conclude that medical applications, they introduce unique security and privacy challenges. The robust and privacy enhancing learning based on recent practical technologies has a high potential to overcome these challenges, and our work has been focused on enhancing the state of the art and defenses and methods to enable secure private learning on sensitive patient data and beyond. None of this would have been possible without my amazing PhD students and postdocs and alumni who have worked on this problem, UCSD Center for Machine-Intelligence, Computing and Security, which I co-lead, and several collaborators that I'm listing here, including collaborators in UCSD Health and inside and outside my own department and generous support from multiple funding sources. Thank you so much for the opportunity. [APPLAUSE]