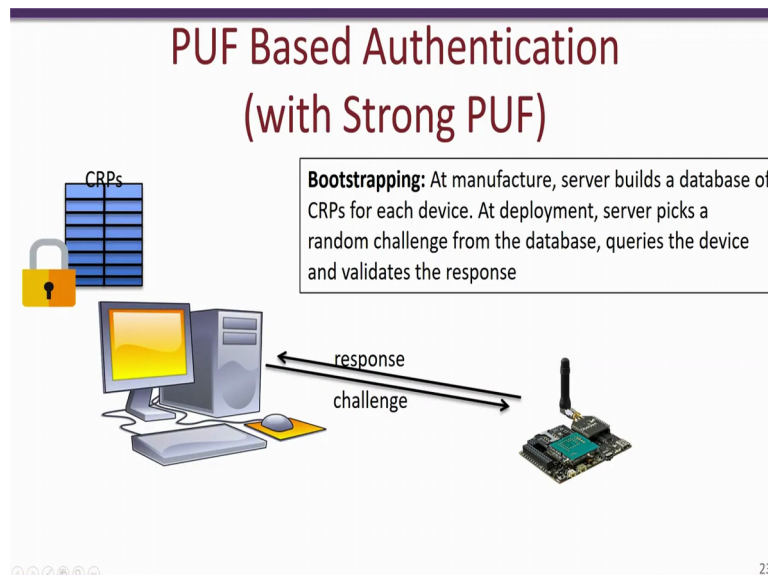


Information Security - 5 - Secure Systems Engineering
Professor Chester Rebeiro
Indian Institute of Technology, Madras
Physically Unclonable Functions (part 3)

Hello and welcome to this 3rd part of physically unclonable functions, this is part of the NPTEL course Secure Systems Engineering. So in the previous 2 video lectures we had an introduction to physically unclonable functions and we had seen that it is essentially a technique digital fingerprinting technique that is used to achieve things like authentication without using secret keys stored in a device. We have seen that each PUF needs to have a good intra and inter-chip variation for it to be actually useful for cryptographic purposes like authentication. In this lecture we will be looking at the use of PUFs to provide authentication.

(Refer Slide Time: 1:03)



So the thing what we will be actually looking at a setup where we have a server over here and we have an edge device and this edge device has a physically unclonable function that is PUF present in it. So the problem that we are actually trying to solve here is that how would the server authenticate this particular device using the PUF. So the way to go about it is that at the time of manufacture of this PUF, the manufacturer would create a database for challenges and the corresponding responses, so this database over here is known as the CRP database and it would be stored in the server. So the CRP database contains a challenge and the expected response from this particular device.

So when this edge device is fielded and it is actually used in in some applications at some time or the other the server would require that this device needs to be authenticated. When

such authentication is required, the server would pick up a challenge from the CRP table and send this particular challenge to this edge device. The edge device would then use this challenge in its PUF and often the corresponding response, so that response is sent back to the server. So note that since we are assuming that there is no cryptography present, there is no encryption or decryption or any other stored keys present in the edge device therefore, the challenge and the response would be sent in clear text.

Now once the server obtains the response from the edge device, it would look of the CRP database that it has stored and it would compare the CRP the response stored in the database with the response obtained from the edge device. Essentially it would look at the Hamming distance between the two and determine whether this response has actually originated from this specific device so in this way the edge device will be authenticated. Now for instance let us say that there is another robe device that is present here and which is trying to masquerade as the original edge device. Now when the server sends the challenge, the properties of the PUF would make it difficult to predict what the response would be further, even if the PUF is implemented in this robe device the response will look quite different than what the original response was from the correct edge device.

Therefore when the response goes back to the server, the server would be able to identify that this response does not match the response stored in the database and therefore it would reject the device, it would say that the authentication is not successful, so one aspect that is an issue with PUF-based authentication technique is the case of the man in the middle. So note that we said that the challenge and the response is sent in clear text and therefore any man in the middle could view the challenge and the corresponding response. Now if the server actually sent the same challenge again, the man in the middle the attacker would be able to actually respond to that corresponding challenge without actually forwarding the challenge to the device.

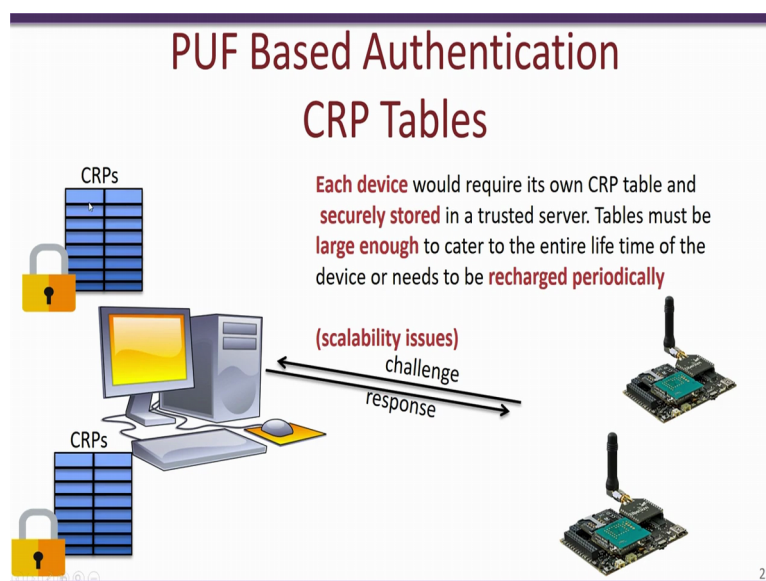
So in order to prevent this man in the middle attacks on PUFs, what is required is that the CRP once used should not be used again which means that once the server actually uses challenge and obtains the corresponding response that particular entry should be marked as the activated or removed from these CRP database are never used again, so this way the man in the middle attack could be prevented to a far extent. One negative aspect of having the man in the middle attack and actually preventing the reuse of CRPs is the fact that the side of the

CRP database should be extensively large, the reason for this is that the CRP tables are generally created at the time of manufactured or before the device is filled.

So therefore, for the entire lifetime of this particular edge device we should have sufficient amount of challenge and corresponding responses stored in the server so that the periodic authentication is supported for the entire life. For example, let us say that we have this edge device which is a put in a remote power plant and this edge device is expected to be used for say let us say for 3 years. Further, we assume that every day there should be challenged and response sent from the server to this edge device so as to authenticate the edge device, this would mean that the CRP database should have over thousand different challenges and response corresponding to this single edge device.

So the periodicity of the authentication would vary from application to application, it could be much smaller than authenticating a single date so there could be application where you require authentication every say 45 minutes or so and therefore you would end up with very large CRP tables.

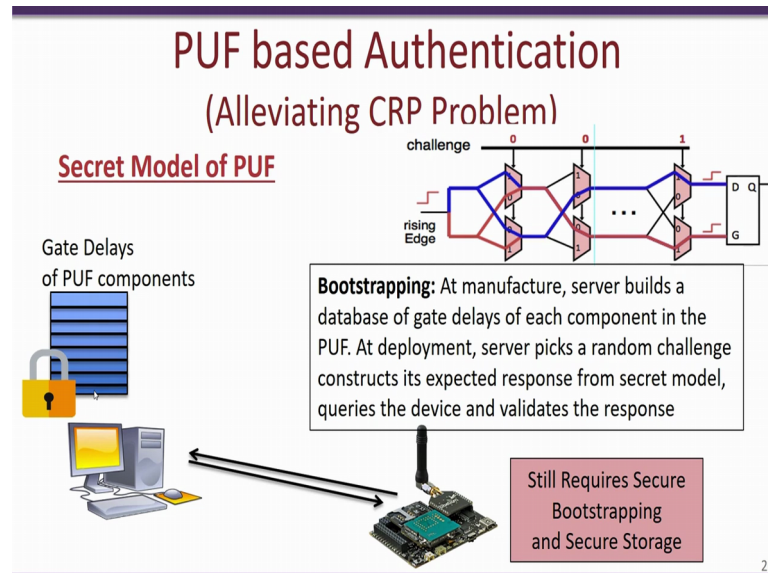
(Refer Slide Time: 7:04)



Further, what we also see is that each CRP table is device specific and this is true because each CRP response to the unique challenge and responses corresponding to each device. Therefore now things get much more worse because if we have multiple devices which are managed by a single server, there would be multiple such CRP tables that are required to be stored in the server database. Other aspects that could be an issue is that the CRP tables if they are stolen or if the privacy of the server gets breached then the entire security of the

PUFs corresponding to these devices would be lost. So researchers have been trying several alternate techniques where they could either reduce the size of the CRP tables or alternatively try to eliminate the use of CRP tables in the authentication process when PUFs are used.

(Refer Slide Time: 8:11)

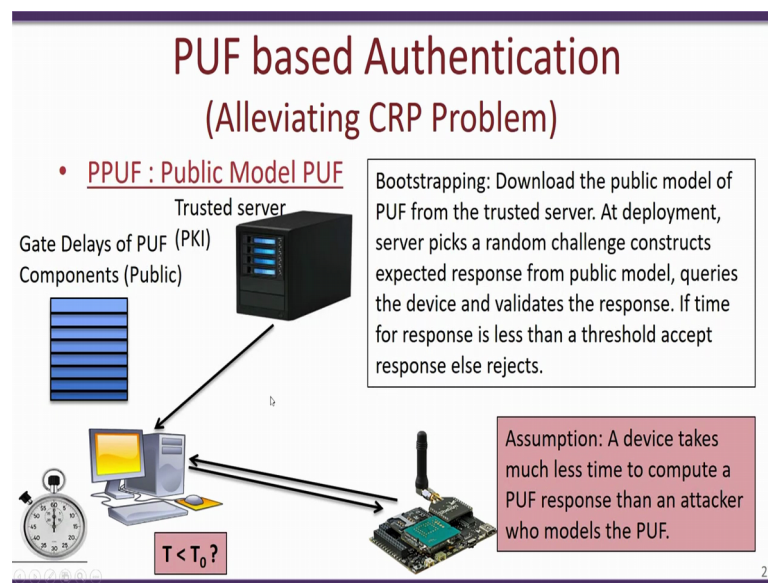


So one very PUFular model is something known as the secret model of PUF. So what happens over here is at the time of manufacture instead of wearing the device with different challenges of training the responses and storing the challenge response pairs in our database, what happens in a secret model PUF is that the manufacturer would study the properties of this PUF and create a model for that particular PUF. For example, the server could build a database of the various gate delays that are present in this arbiter PUF and it would actually required to store this model for this specific arbiter PUF. So what this model would be actually capable of doing is that given a particular challenge this particular model could create a very good estimate of what the response for a particular PUF should be for that specific challenge.

So what is required is that this model for the PUF is kept secret and then the device is actually fielded and when authentication is required, the server would randomly choose a particular challenge, it would use this model of this particular PUF to create what is the expected response for that particular challenge. It would then send out the challenge to this device and obtain the response; it would then compare this response to what is the expected response obtained from the model, close match between the expected response and the actual response obtained from the device for that particular challenge would then be used to authenticate the device.

Now this works quite well, especially on PUF switch and we actually modelling such a way where the secret model of this PUF can be extracted at the manufactured time but nevertheless this technique still has a limitation. The limitation is that this PUF model still has to be kept secret and it has to be extremely secret because of this model is lost then any attacker could snoop into the challenge that is sent from the server to that edge device, use that model which it has just stolen and provide the response was that particular challenge. In this may be attacker machine can get authenticated without actually having a PUF.

(Refer Slide Time: 10:40)



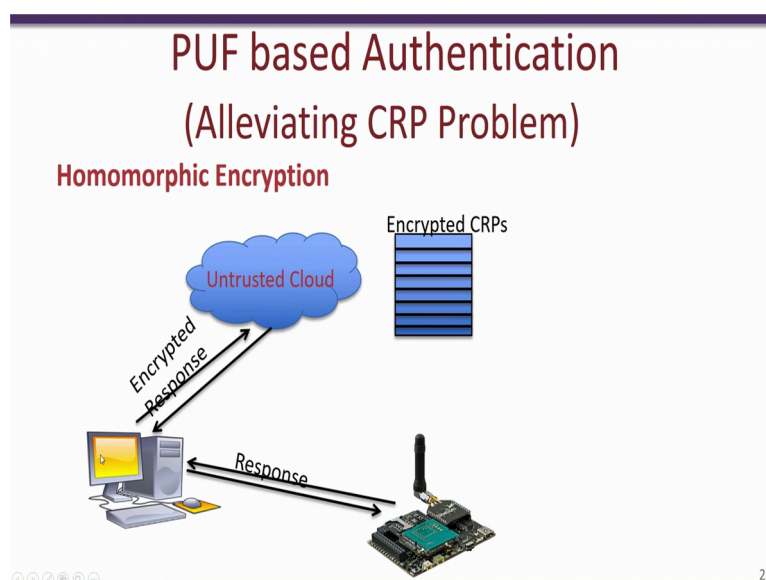
So there are being other works such as the public model PUF which has been researched. So this public model PUF works in a very similar way, so except for the fact that the model for the PUF which is developed using the various gate delays and stored in the database present in the server does not have to be secret. So the fact that is actually use over here is that is for a particular challenge that is sent to the actual device that owns the right PUF, obtaining the response will just take a few nanoseconds therefore, if a challenge is sent to the device which actually has the PUF the server would obtain the response very quickly.

On the other hand, if that is an attacker who actually has a copy of this particular model of the PUF, sending the challenge and computing the model based on this public model would take quite some time even with heavy computing resources. Therefore, the delay between the sending the challenge and obtaining the response from an attacker device would be considerable. Therefore in this form of authentication what is suggested is that the attacker picks out a random challenge, uses this public model for the PUF to obtain what an expected

response for that particular challenge and then sends out the challenge to the device. It also notes the time taken for the response to be obtained.

Now if the time taken is greater than some particular threshold in this case T_{not} , then it is assumed that the response is obtained from malicious device or from an attacker device and is ignored and no authentication is done. On the other hand, if the time to update the response is very short much lesser than the threshold then the server would validate the response with the expected response obtained from this model of the PUF, and if the match is quite closed to this to the expected response than the device would get authenticated. While this particular technical fields that other factors like network delays and routing delays and so on are not considered, this technique would work quite well for small systems with small local networks.

(Refer Slide Time: 13:13)

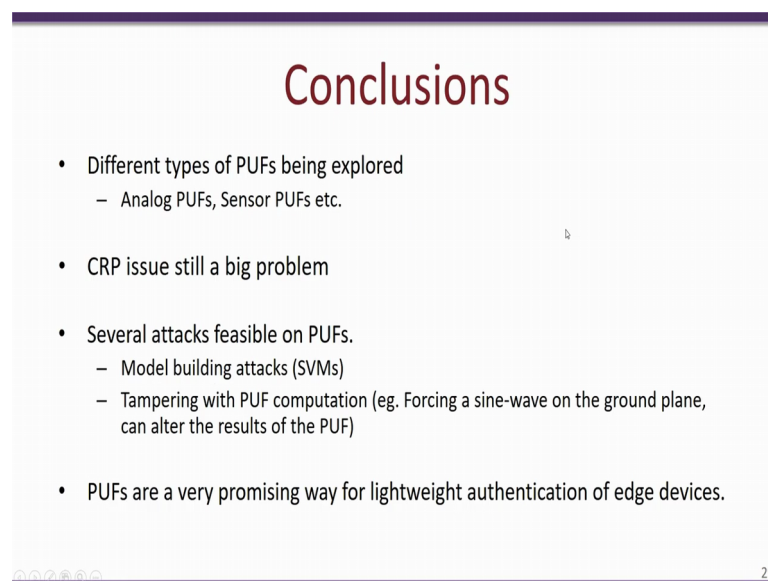


Another technique where there is a considerable amount of research going on is to use something known as homomorphic encryption. Using this, the challenge and response pairs which is generated for this particular PUF present in the device is encrypted and stored in an un-trusted environment. Now the basic property of using homomorphic encryption is the fact that the validation of the particular response can be done on encrypted data. This means that the server could actually run a particular program in this un-trusted cloud which works on the encrypted CRP database and would be able to actually obtain whether the response is in fact valid or not valid even without decrypting the CRP database.

So in this particular model as usual, the server would actually send the particular challenge to this edge device, obtain the corresponding response which if valid is often from the PUF and this response is then sent to the un-trusted and environment, this would be a regular cloud computing environment, the homomorphic encryption technique used present in this I trusted and environment than works on this and created database and validate whether the response is indeed the response which is stored in the database. In this way even though this cloud is un-trusted the security of the database is ensured because of the homomorphic encryption present.

Now this seems like a very good solution for solving CRP problem in PUF, there are still a lot of research before actually taking this homomorphic encryption to practice. The reason being that it takes still considerable amount of time to actually validate and scripted responses using an encrypted database although a lot of research is actually taking place in order to reduce this time requirements.

(Refer Slide Time: 15:15)



Conclusions

- Different types of PUFs being explored
 - Analog PUFs, Sensor PUFs etc.
- CRP issue still a big problem
- Several attacks feasible on PUFs.
 - Model building attacks (SVMs)
 - Tampering with PUF computation (eg. Forcing a sine-wave on the ground plane, can alter the results of the PUF)
- PUFs are a very promising way for lightweight authentication of edge devices.

29

To conclude the video lectures on PUF, PUFs are extremely useful techniques or mechanisms to achieve various cryptographic things like authentication, secret key generation and so on. And with a very small fingerprint and also it is ensured that if a PUF is present in a device then that particular device cannot be cloned. There is a lot of research which is going on to find actually new PUFs such as analog PUFs, PUFs using sensor and so on. There is still a huge problem of solving the CRP issue and how the CRP tables could be actually compressed so that the efficiency and the memory storage can be reduced.

One of the major drawbacks of PUFs which is preventing it quite a bit from actually going to commodity devices is these attacks known as model building attacks. So within a model building attacks what happens is that you could consider a scenario where you have a server and a device, and this device has a PUF and the server over a period of time is sending challenges and obtaining response. Now there is also a passive listener in this entire thing who views these challenges and the responses and over a period of time uses machine learning techniques or model building technique to build a model of that PUF.

Now after a certain number of authentications have completed, for a given challenge the attacker could use the model for that PUF which it has actually created which it has actually built and respond to the challenge. Thus you see that authentication will break in this way because the attacker without actually owning the current PUF would be able to provide the right responses for challenges. Another big problem with PUF is that of tampering with a computation for example, during a PUF computation is for instance an attacker is able to actually get hold of the device and manipulate the device operation by say forcing sinusoidal waves in the power or the ground plane then the response from the PUF can be altered.

So if the PUF response is altered beyond a particular degree, the authentication would fail because the server would find a large amount of mismatch with the response which is stored in the CRP table and the response of obtained by the PUF hardware, this would be more like a denial of service attack, where the attacker rather than learning what the response would be is essentially preventing the device from getting authenticated. Nevertheless PUFs are very promising way for lightweight authentication of its devices and perhaps in the near future we would actually see a lot of forms being used in these devices and this would ensure that the devices will not get cloned, no secret key is required in the device and so on, thank you.