PUF(Physical Unclonable Functions)

What’s a PUF?

PUF ‘s are circuits that produce unique signature bits based on process variations that occur in manufacturing process. By using PUF’s we don’t need to buy expensive cryptographic algorithm for security instead we can use the hardware itself as the security device.

We also don’t need to store the secret keys in digital memory such as SRAM and EEPROM which could be pervasive to many attacks. Now, we will look into how puf works in authentication and key generation.

**Previous Works on Unique objects**:

PUF has been made to test that a similar PUF cannot be made from the existing PUF or can be replicated. Since its unclonable, we say the PUF circuit as the “fingerprint” of the security devices.

**Classification of PUF**:

PUFs are modelled as CRP’s (Challenge Response Pairs), which means that when we give a challenge(c) to the PUF we get a response(r). So, depending on the challenge we can we get different response. The mathematical representation is given as:

r = f(c), where the function represents the unique property of PUF.

Broadly PUFs can be classified into two categories:

**Weak PUF**: Used For key storage. A weak PUF (also called as Physically Obfuscated PUF’s) has a very small range such that we can have collisions with one or other challenges that give the same response. We define the Weak PUF function as r=F(.) where F(.) means it has only very small domain. Even though it can be used as a fingerprint to generate cryptographic functions, the number of responses of the weak PUF is related to the number of components subject to manufacturing variation. A weak PUF can have only some secret keys (i.e the domain range is less).

**Strong PUF**: Commonly used for authentication. Strong PUf’s have a large domain and supports a lot of CRP’s that means it can be used effectively and securely.

**Error Correction**:

We know that circuits response changes not only because of challenge but also because of the temperature and other environmental conditions. So, these affects the digital signature of the circuit to fail.

To tackle these problems, we use **differential design techniques** to filter the first order environmental dependencies. To improve the performance of the PUF we use soft coding technique and for reliability we make the PUF to give repeated outputs to get a stable output.

**Strong PUF Architectures:**

**OPTICAL PUF:**

Optical PUF was found by pappu et al. he found three components:

1) a laser directed along the Z-axis that can be moved in the XYplane

and whose polarization can be modified.

2) a stationary scattering medium that sits along the path of the laser beam.

3) an imaging device that records the

output ‘‘speckle’’ pattern of laser light exiting the scattering

medium.

Challenge: laser XY Location and polarization.

Response: Speckle pattern.

**Authentication of a Strong PUF**:

The following steps are used for authentication of a PUF.

1. PUF of its own fingerprint is manufactured.
2. Server generates all possible CRPS’s and stores them as a secret message.
3. PUF is given to a client.
4. Client submits request to the server.
5. Server picks a known CRP and submits the challenge to the client.
6. The client runs the challenge on the PUF and returns the response to the server.
7. Server checks to see that the response is correct and marks the CRP as used.

When we use a challenge, the server must make sure that the same challenge is not used again.

**Intra PUF Variation**:

Its defined as the number for bits in a PUF response that change when the same challenge input is given to the PUF. Usually, this happens due to environmental variation and statistical noise.

**Inter-PUF Variation**:

Its defined as the number of bits in a PUF response that vary between different devices for the same challenges. This is due to differences in IC Fabrication. It’s the measure of the uniqueness

of a PUF circuit.

A good PUF must have low values for Intra PUF Variation and a high value for Inter PUF variation.

**Weak PUF Architecture**:

**Ring Oscillator PUF**:

The design of a Ring Oscillator PUF has a not gate with 4 Not Gates connected with each other and the output is given as a feedback to the input.

We concatenate all Ring oscillators and use a decoder to select one oscillator at a time to use that intrinsic variability of that ring oscillator PUF.

**SRAM PUF**:

The SRAM has two states (either 0 or 1 state) , these state values differ when the SRAM is powered on because it displays the initial values which are in the SRAM those SRAM values are used to generate the PUF design.

**KEY GENERATION IN WEAK PUF’S**:

The cryptographic key that is derived from the weak PUF , once its derived the weak PUF its stored in the secure volatile memory for device’s operation.

PROTOCOL:

The key generated by weak PUF is used for authentication. By supplementing the weak PUF with hardware HMAC/AES implementation we can achieve authentication capability that embody the HMAC/AES protocol.

EMERGING PUF CONCEPTS:

Although existing PUF technology has been successful in addressing applications in low cost authentication and secure key generation, PUF technology still has significant untapped potential.

**MODEL BASED PUF’S**:

While considering the application of low cost authentication, one of the primary drawbacks of strong PUF architectures is the establishment of the secret message challenge. This secret model PUF still requires both the “secure bootstrapping” phase as well as the secure storage. This model must be kept secret, as it exactly describes the PUF behavior and this can be spoofed in the authentication process. So, we can directly compute the random challenge and computing the response.

**TIMED AUTHENTICATION AND PUBLIC MODELS**:

As the model of the PUF has large storage requirement so we go to new type of concept on PUF called as timed authentication PUF’s, PPUF’s, SIMulation Possible but Laborious (SIMPL) systems.

A PPUF has a model that is public (i.e known to everyone). The authentication scheme works as follows (Where a server is authenticating a client):

1. Server obtains the desired PPUF model from a trusted third-party storage.
2. Server generates a challenge and computes the response using the PPUF model.
3. Server sends response to the client and begins the timer.
4. The client uses its PPUF hardware to compute a response and send it back to the server.
5. Server measures the client response time T.
6. Server accepts if T<T0 and client’s response is correct.

Eventhough the PUF model is public, the model is resistant against tampering. This can be done using the traditional public key infrastructure(PKI). The server must establish a value T0 as described in the above mechanism.

1. The time must be greater than the PPUF hardware to compute the response and allow for roundtrip network latency.
2. Short enough that no model could emulate the PPUF hardware and correctly produce a response in that time.