A Modeling Attack Resistant Deception Technique for Securing PUF based Authentication

Chongyan Gu*, Chip Hong Chang†, Weiqiang Liu‡, Shichao Yu*, Qingqing Ma‡ and Máire O'Neill*
*Centre for Secure Information Technologies (CSIT), ECIT
†Queen's University Belfast, Belfast, U.K. {cgu01, syu08}@qub.ac.uk, m.oneill@ecit.qub.ac.uk
†School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore, echchang@ntu.edu.sg
‡College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics (NUAA)
Nanjing, China, liuweiqiang@nuaa.edu.cn

***Abstract*—Due to practical constraints in preventing phishing through public network or insecure communication channels, simple physical unclonable function (PUF)-based authentication protocol with unrestricted queries and transparent responses is vulnerable to modeling and replay attacks. In this paper, we present a PUF-based authentication method to mitigate the practical limitations in applications where a resource-rich server authenticates a device with no strong restriction imposed on the type of PUF designs or any additional protection on the binary channel used for the authentication. Our scheme uses an active deception protocol to prevent machine learning (ML) attacks on a device. The monolithic system makes collection of challenge response pairs (CRPs) easy for model building during enrollment but prohibitively time consuming upon device deployment. A genuine server can perform a mutual authentication with the device at any time with a combined fresh challenge contributed by both the server and the device. The message exchanged in clear does not expose the authentic CRPs. The false PUF multiplexing is fortified against prediction of waiting time by doubling the time penalty for every unsuccessful authentication.**

***Index Terms*—Physical Unclonable Function (PUF), machine learning attacks, authentication protocol.**

I. INTRODUCTION

A physical unclonable function (PUF) is a security primitive that utilizes intrinsic manufacturing process variations to generate a unique digital fingerprint that is inherently difficult to clone and tamper-evident [1]. Since the initial proposal to leverage on Hamming distance (HD) threshold for lightweight PUF-based authentication [2], similar PUF-based authentication protocols have been derived to endow linearly-sized Strong PUF (SPUF) circuits with exponentially large CRP capacity for device authentication. Unfortunately, SPUFs used for device authentication with limited nonlinear mixing have been shown to be vulnerable to ML based modeling attacks. Masquerade attacks can be perpetrated through malicious nodes and unprotected communication links of ad hoc networks to efficiently collect a large number of CRPs to model the PUF.

A survey on entity authentication protocols using PUFs [3] shows that most PUF protocols [4]–[8] are heavyweight or require complicated protocol operations that add to the hardware implementation area, power and performance overheads. Only two protocols, the slender PUF [9] and noise bifurcation [10], require neither an error correction code (ECC) nor a strong cryptographic algorithm but a true random number generator (TRNG) to provide heuristic security against modeling attacks. The idea of slowing down the read out of PUF response was first raised in [11]. The SHIC PUF [11], implemented by novel semiconductor technology, does not integrate well with conventional CMOS designs [12]. The incompatibility raises constraints on the nano-micro link. As SHIC is a weak PUF, its CRP space grows only linearly as opposed to exponentially with array size. Recently, Yu *et al.* [12] proposed two lockdown techniques to limit the number of authentication requests. The first lockdown protocol allows only unilateral authentication while the second lockdown protocol requires a TRNG on the device side to generate a device nonce for bilateral authentication initiated by the server. The responses are sent in clear and a run time check for state segment overlap of linear-feedback shift register (LFSR) is required to prevent reply attack. Lockdown protocols support only a limited number of authentications, and rely on the hardness of XOR PUF to support model-based authentication. The hardware cost of XOR PUF increases and reliability reduces commensurately with increasing number of XORs for higher ML resistance. The most recent work from Gao *et al.* [13] uses a reconfigurable latent obfuscation technique to conceal and distort the relationship between CRPs. The pattern vectors for challenge and response obfuscation are selected by a random number generator (RNG), and are made latent and reconfigurable per authentication session. Nevertheless, the most recent report shows that both methods [12], [13] are vulnerable to the protocol attack [14].

In this paper, a deception based authentication scheme is proposed to allow a server to authenticate a deloyed device at any time without specifically limiting the number of authentications. A monolithic architecture is built around a lightweight machine learnable SPUF at the device side to enable a constant size SPUF model to be built during enrollment and kept at the server side for server initiated bilateral authentication. Upon deployment, the device tracks the time between two successive challenge applications and uses this to deceive or excessively delay the adversary attempting to collect the CRPs by brute-

978-1-7281-3544-1/19/$31.00 2019 IEEE.

Authorized licensed use limited to: Queens University Belfast. Downloaded on July 19,2023 at 14:01:38 UTC from IEEE Xplore. Restrictions apply.

force queries made on the device or protocol interface. A fake SPUF is used to generate false responses to the adversary who uses the enrollment interface to query the device within the waiting period. The fake SPUF is reused for nonce generation at the device side in response to the authentication request made by the server who has the knowledge of its unused CRPs. No response is transmitted in clear in the proposed bilateral protocol to prevent eavesdropping attacks. The waiting time to adversary consists of a static and a dynamic component. The dynamic component escalates on each fake response generation, which rapidly blows up the waiting time on adversarial queries. The false PUF multiplexing is also complemented by the random adjustment of the dynamic waiting time component for each successful mutual authentication by the genuine server to further delude the attacker into using an incorrectly predicted waiting time for model building attack.

II. The Proposed Deception Protocol

To support machine learn resistant model-based authentication, the genuine SPUF ($\mathsf{SPUF^G}$) is monolithically integrated with a fake SPUF ($\mathsf{SPUF^F}$), a pseudo random number generator (PRNG), a L-bit counter, a L-bit register, a comparator and a simple controller as shown in Fig. 1. When an n-bit input challenge c is applied to the device, m sub-challenges are derived from the PRNG using c as a seed. If the select enable strobe S of the multiplexer is 0, these sub-challenges are input to $\mathsf{SPUF^G}$ to generate an m-bit response. Otherwise, the response will either be generated from $\mathsf{SPUF^G}$ or $\mathsf{SPUF^F}$ depending on the comparator output Z. The comparator compares the output of Q with that of P. $Z = 1$ if they are not equal and $Z = 0$ otherwise. Q is cleared upon power-up reset or when $\overline{clr} = 0$. Q will count continuously until it is halt by $Z = 0$. A simple controller is used to manage the inputs and outputs to the external world when the device is queried.

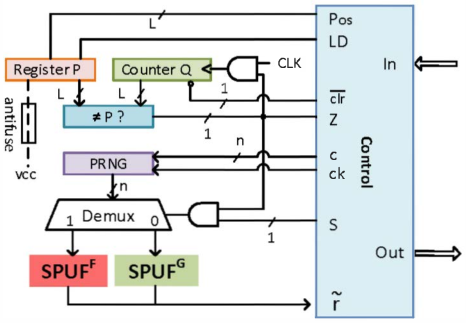

Figure 1. Proposed lockdown encapsulation of genuine SPUF for deception-based authentication.

A. Enrollment

During enrollment, sufficiently large number of CRPs are extracted from $\mathsf{SPUF^G}$ to train an authentication verification model $\mathsf{SPUF^G_{model}}$ by machine learning algorithm for later reproduction of any CRPs of $\mathsf{SPUF^G}$ with a high accuracy. As shown in Fig. 2, after $\mathsf{SPUF^G_{model}}$ has been successfully built, one bit of register P is forced to stick at logic one permanently by an irreversible antifuse or an one-time programmable (OTP)

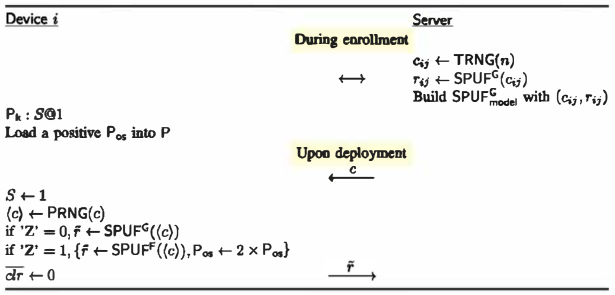
Device i
Server
During enrollment
$c_{ij} \leftarrow \mathsf{TRNG}(n)$
$\longleftrightarrow$
$r_{ij} \leftarrow \mathsf{SPUF^G}(c_{ij})$
Build $\mathsf{SPUF^G_{model}}$ with (c_{ij}, r_{ij})
$\mathsf{P_k}: S@1$
Load a positive $\mathsf{P_{os}}$ into P
Upon deployment
$\xleftarrow{c}$
$S \leftarrow 1$
$\langle c \rangle \leftarrow \mathsf{PRNG}(c)$
if 'Z' $= 0, \tilde{r} \leftarrow \mathsf{SPUF^G}(\langle c \rangle)$
if 'Z' $= 1, \{\tilde{r} \leftarrow \mathsf{SPUF^F}(\langle c \rangle), \mathsf{P_{os}} \leftarrow 2 \times \mathsf{P_{os}}\}$
$\overline{clr} \leftarrow 0$
$\xrightarrow{\tilde{r}}$

Figure 2. Direct query for model-building during enrollment and after device deployment.

non-volatile memory (NVM) bitcell. Upon device deployment, if the same protocol is used by the adversary to query the system, the response will only be output from $\mathsf{SPUF^G}$ if the challenge is input after a duration of $\mathsf{T_{min}}$, where $\mathsf{T_{min}} = 2^k \times \mathsf{T_{clk}}$, k is the stuck-at-one bit of P. To prevent the adversary from modelling $\mathsf{SPUF^G}$ after device deployment, k and $\mathsf{T_{clk}}$ are designed such that it is impossible for an adversary to collect a minimum number of CRPs from $\mathsf{SPUF^G}$ to accurately model $\mathsf{SPUF^G}$ in a practical amount of time. $\mathsf{T_{min}}$ can be adapted to the machine-learn resistance of the chosen $\mathsf{SPUF^G}$ and the use-case. Even with the same number of CRPs required for a successful attack on a selected SPUF, $\mathsf{T_{min}}$ can be set differently according to the application risk due to the different service span and diminishing profitability of a successful attack with time.

B. Deception-based Technique Against Model-building Attack

Fig. 2 shows the devices action when the enrollment protocol is utilized by the adversary to collect the CRPs for model building. When the device is queried by a challenge c, the device sets $S = 1$ and reads the comparator output Z. If $Z = 0$, the time lapse T of the current query from the last output response is at least $\mathsf{T_{min}}$, this frequency of authentication events is perceived to be normal with enough safety margin against model building attack. The controller will apply the sub-challenges $\langle c \rangle$ derived from c to $\mathsf{SPUF^G}$ to produce a response $\tilde{r}$. When $\mathsf{T} = \mathsf{Q} \times \mathsf{T_{clk}} < \mathsf{T_w}$, where $\mathsf{T_w} = \mathsf{P} \times \mathsf{T_{clk}}$, then $Z \neq 0$ signifies that the authentication events are unusually frequent. In this case, a response $\tilde{r}$ from a fake SPUF ($\mathsf{SPUF^F}$) will be output by applying $\langle c \rangle$ to it, followed by clearing Q with $\overline{clr} = 0$ and doubling $\mathsf{P_{os}}$. $\mathsf{P_{os}}$ is a nonzero integer that can be loaded into P by the server to extend $\mathsf{T_w}$ beyond $\mathsf{T_{min}}$ through the proposed bilateral authentication protocol described in the next subsection.

As each consecutive query within the waiting time $\mathsf{T_w}$ will double $\mathsf{P_{os}}$, the waiting time will extend rapidly. Hence, an adversary who try to brute-force attack the SPUF system using the enrollment interface will receive fake responses from $\mathsf{SPUF^F}$. Using the incorrect CRPs to train the model will result in either non-convergence or convergence with highly inaccurate prediction results. Unknowingly using such an incorrect SPUF model to mount an attack on the target device will easily expose the adversary. Comparing with output

Authorized licensed use limited to: Queens University Belfast. Downloaded on July 19,2023 at 14:01:38 UTC from IEEE Xplore. Restrictions apply.

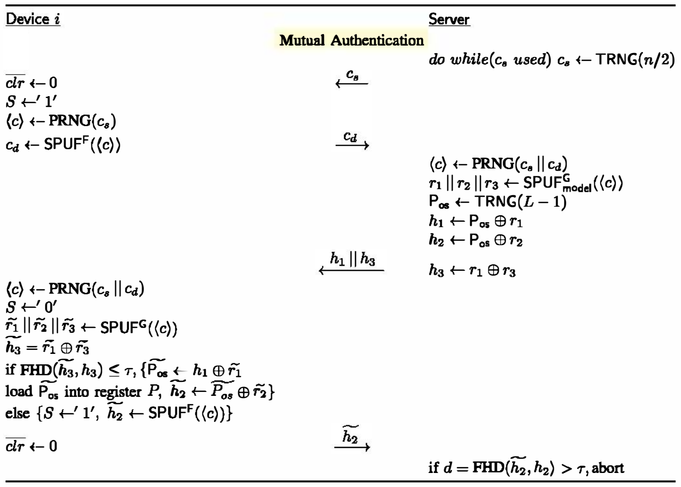
Device i
Server
Mutual Authentication
$do\ while(c_s\ used)\ c_s \leftarrow \mathsf{TRNG}(n/2)$
c_s
$\overline{clr} \leftarrow 0$
$S \leftarrow' 1'$
$\langle c \rangle \leftarrow \mathrm{PRNG}(c_s)$
$c_d \leftarrow \mathsf{SPUF}^{\mathsf{F}}(\langle c \rangle)$
c_d
$\langle c \rangle \leftarrow \mathrm{PRNG}(c_s \,||\, c_d)$
$r_1 \,||\, r_2 \,||\, r_3 \leftarrow \mathsf{SPUF}^{\mathsf{G}}_{\mathsf{model}}(\langle c \rangle)$
$\mathsf{P_{os}} \leftarrow \mathsf{TRNG}(L-1)$
$h_1 \leftarrow \mathsf{P_{os}} \oplus r_1$
$h_2 \leftarrow \mathsf{P_{os}} \oplus r_2$
$h_1 \,||\, h_3$
$h_3 \leftarrow r_1 \oplus r_3$
$\langle c \rangle \leftarrow \mathrm{PRNG}(c_s \,||\, c_d)$
$S \leftarrow' 0'$
$\widetilde{r_1} \,||\, \widetilde{r_2} \,||\, \widetilde{r_3} \leftarrow \mathsf{SPUF}^{\mathsf{G}}(\langle c \rangle)$
$\widetilde{h_3} = \widetilde{r_1} \oplus \widetilde{r_3}$
if $\mathrm{FHD}(\widetilde{h_3}, h_3) \leq \tau$, $\{\widetilde{\mathsf{P_{os}}} \leftarrow h_1 \oplus \widetilde{r_1}$
load $\widetilde{\mathsf{P_{os}}}$ into register P, $\widetilde{h_2} \leftarrow \widetilde{P_{os}} \oplus \widetilde{r_2}\}$
else $\{S \leftarrow' 1',\ \widetilde{h_2} \leftarrow \mathsf{SPUF}^{\mathsf{F}}(\langle c \rangle)\}$
$\overline{clr} \leftarrow 0$
$\widetilde{h_2}$
if $d = \mathrm{FHD}(\widetilde{h_2}, h_2) > \tau$, abort

Figure 3. Proposed deception-based mutual authentication protocol flow.

nothing or a fixed response to alert the attackers, the false PUF responses waste the attacker's time and resources to build an incorrect model. This will increase their opportunity cost to attack another target or switch attack strategy for a better chance of success.

The server with $\mathsf{SPUF}^{\mathsf{G}}_{\mathsf{model}}$ can still successfully authenticate the deployed device as frequently as necessary using the proposed mutual authentication protocol presented in the next subsection.

C. Mutual Authentication

The genuine server can use the mutual authentication protocol shown in Fig. 3 to authenticate the device at any time, and change the current waiting time $\mathsf{T_w}$ by changing $\mathsf{P_{os}}$. The minimum waiting time $\mathsf{T_{min}}$ is determined by $\mathsf{T_{clk}} \times 2^k$, where $k \in [0, L-1]$ is the bit position of P that has been forced into the stuck-at-one state after enrollment and $\mathsf{T_{clk}}$ is the clock period of Q. Therefore, the range of waiting time $\mathsf{T_w}$ to apply the next challenge to $\mathsf{SPUF}^{\mathsf{G}}$ can be varied from $\mathsf{T_{min}}$ to $\approx 2^{L-k} \times \mathsf{T_{min}}$ by loading the remaining $L-1$ bits of the register with a non-zero integer $\mathsf{P_{os}}$. The maximum offset time $\mathsf{T_{os}}$ that can be added to $\mathsf{T_{min}}$ ranges from $2^{L-1} \times \mathsf{T_{clk}} \approx \frac{1}{2}\mathsf{T_{min}}$ for $k = L-1$ (i.e., the MSB of the register) to $(2^L - 2) \times \mathsf{T_{clk}} = (2^L - 2) \times \mathsf{T_{min}}$ if $k = 0$ (i.e., the LSB of the register). By adjusting $\mathsf{T_{os}}$ along with this bilateral authentication, it becomes more difficult for an adversary to estimate the minimum $\mathsf{T_w}$ to collect enough valid responses by brute-force query as the waiting time will escalate with every incorrect attempt.

To initiate a bilateral authentication, the server randomly selects an unused half-length challenge c_s and sends it to the device. Upon receiving c_s, the device uses it as a *seed* to its PRNG to generate $n/2$ random sub-challenges $\langle c \rangle$. The device then clears the counter to ensure that $Z = 1$ so that $\mathsf{SPUF}^{\mathsf{G}}$ or $\mathsf{SPUF}^{\mathsf{F}}$ can be selected by S to receive $\langle c \rangle$. By setting $S = 1$, these sub-challenges are applied to $\mathsf{SPUF}^{\mathsf{F}}$ to produce a half-length challenge c_d to the server. The process is aborted by the server if c_d has been used. Otherwise, the server concatenates c_s and c_d to complete the full-length challenge $c_s \,||\, c_d$ and uses it as a *seed* to the same PRNG as the device to generate $3m$ sub-challenges $\langle c \rangle$ to produce three m-bit responses, r_1, r_2 and r_3 from $\mathsf{SPUF}^{\mathsf{G}}_{\mathsf{model}}$. Two helper data $h_1 = \mathsf{P_{os}} \oplus r_1$ and $h_3 = r_1 \oplus r_3$ are computed, where $\mathsf{P_{os}} = \mathsf{T_{os}}/\mathsf{T_{clk}}$ is a $(L-1)$-bit positive integer. If $m \geq L$, the m-bit response is truncated to match the length of $\mathsf{P_{os}}$ before the xor operation. h_1 and h_3 are concatenated into $h_1 \,||\, h_3$ and sent to the device.

Upon receiving the $(m+L)$- bit string, using the same sub-challenges $\langle c \rangle$ with $c_s \,||\, c_d$ from its PRNG, the device generates three m-bit responses, $\widetilde{r_1}$, $\widetilde{r_2}$ and $\widetilde{r_3}$, by setting $S = 0$ to select $\mathsf{SPUF}^{\mathsf{G}}$ for the application of $\langle c \rangle$. Then it computes $\widetilde{h_3} = \widetilde{r_1} \oplus \widetilde{r_3}$. If $\widetilde{h_3}$ matches the received h_3 within acceptable fractional Hamming distance (FHD) tolerance, the server is authenticated. The device then recovers $\widetilde{\mathsf{P_{os}}}$ from the received h_1 by XORing it with $\widetilde{r_1}$.

The recovered $\widetilde{\mathsf{P_{os}}}$ is loaded into the register to change the waiting time to $\mathsf{T_w} = \mathsf{T_{min}} + \widetilde{\mathsf{T_{os}}}$. The device will acknowledge the successful update by sending $\widetilde{h_2}$ to the server, where $\widetilde{h_2} = \widetilde{\mathsf{P_{os}}} \oplus \widetilde{r_2}$. Otherwise, if the FHD between $\widetilde{r_3}$ and r_3 exceeds the acceptable tolerance, the device sets $S = 1$ to generate $\widetilde{h_2} = \mathsf{SPUF}^{\mathsf{F}}(\langle c \rangle)$ and sends it to the server. Q is cleared by setting $\overline{clr}$ upon transmission of $\widetilde{h_2}$. The server can verify the authenticity of the device by checking the received $\widetilde{h_2}$ against $h_2 = \mathsf{P_{os}} \oplus r_2$. If they are equal within an acceptable FHD tolerance, the device is authenticated successfully. Otherwise, the authentication fails and the process is aborted.

It should be noted that the adversary cannot issue an unseen packet $h_1 \,||\, h_3$ that has not been issued by the server to obtain the corresponding response packet $\widetilde{h_2}$. Consequently, the uniqueness of every authentication is assured by the fresh challenge. Because r_1, r_2 and r_3 are generated from the same fresh starting challenge and locked to each other by design, replaying either half challenge will not produce new response. Even if $\widetilde{r_1} \neq r_1$ due to the reliability of $\mathsf{SPUF}^{\mathsf{G}}$, the minimum waiting time is still guaranteed by $\mathsf{T_{min}}$. There is no need for the server to keep track of the waiting time. As long as the error due to the recovered $\widetilde{P_{os}}$ and the response $\widetilde{r_2}$ generated by $\mathsf{SPUF}^{\mathsf{G}}$ does not cause the FHD between $\widetilde{h_2}$ against h_2 to exceed τ, the outcome of the authentication will not be affected. Since we do not limit the number of challenges like [12], and there is no waiting time for a genuine server, the server can authenticate the same device again with a fresh challenge before rejecting the device.

III. Results and Analyses

Both covariance matrix adaptation evolution strategies (CMA-ES) and linear regression (LR) attacks are used to evaluate the proposed protocol. We follow the approaches in [15]–[19] to simulate the attacks on Arbiter PUF (APUF). To test the impact of noise on the proposed deception protocol, in all our experiments, a random variable $\sim \mathcal{N}(0, \sigma^2_{noise})$ is inserted into each delay path, $\Delta(n)$, where $\sigma_{noise} = 0.5$ is derived from the practical noise level obtained in [20].

Authorized licensed use limited to: Queens University Belfast. Downloaded on July 19,2023 at 14:01:38 UTC from IEEE Xplore. Restrictions apply.

A. *Effect of False PUF Multiplexing on Modeling Attacks*

The fake response generator is implemented by a SPUF design with a different circuit architecture from the genuine SPUF to produce vastly different CRPs. A RNG function from MATLAB is also used to generate random responses for comparison. The resistance of the deception protocol, $\mathsf{SPUF^G}$ + $\mathsf{SPUF^F}$ are evaluated by both LR and CMA-ES attacks and compared against that of $\mathsf{SPUF^G}$ + RNG in this section.

1) Percentage of Valid/Invalid Responses on LR Attack: Two conventional APUFs, one for $\mathsf{SPUF^G}$ and the other for $\mathsf{SPUF^F}$, are employed to produce the CRPs for training. Three different sizes of CRP sets, $N_{CRP} = 640$, $N_{CRP} = 24,000$ (similar to [15], [16]), and $N_{CRP} = 240,000$ are used for training. Depending on the percentage of fake information, x_f ($x_f \in \{0, 100\}$), a group of mixed response bits from both $\mathsf{SPUF^G}$ and $\mathsf{SPUF^F}$/RNG is collected for the LR attack.

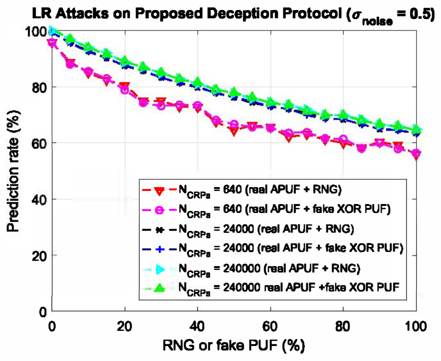

Figure 4. Comparison of LR attack results for the proposed deception protocol utilizing either a RNG or a XOR PUF as fake response generator. The y-axis shows the achieved correct prediction rate P_{pred} of the LR attacks based on different percentages of fake information mixed with the training responses.

Fig. 4 depicts the LR attack results of the proposed deception protocol by using either $\mathsf{SPUF^F}$ or RNG as its fake response generator. A varying number of fake CRPs is produced according to x_f. The prediction rate of modeling attack is calculated by $P_{pred} = \frac{N_{correct}}{N_{total}} \times 100\%$, where $N_{correct}$ and N_{total} are the number of correctly predicted response bits and the total number of response bits, respectively. Ideally, $P_{pred} = 50\%$ for random guess of a binary variable. A successful model should have a P_{pred} significantly larger than 50%. It can be seen that P_{pred} decreases proportionally with the percentage of fake responses x_f in the training samples. For the same x_f, the prediction rate is only slightly higher for a larger number of training samples, *e.g.*, 24,000, than a smaller number, *e.g.*, 640. Increasing the size of the training samples to 240,000 does not yield much difference in the prediction rates. For the same number of training CRPs and the same percentage of fake responses, the prediction rates are the same for both fake APUF and RNG. The reasons for the approximately 60% prediction rate when 100% RNG or fake PUF responses are used for training can be explained as follows. Firstly, the CRPs of the real PUF may not be exactly independently and identically distributed. This can cause a deviation of its prediction rate from the ideal 50%. Secondly, noise ($\sigma_{noise} = 0.5$) has been deliberately added to both the real and fake PUFs to simulate an APUF in practice. This may also contribute to the skew of prediction accuracy when the model is trained by all fake responses.

2) Percentage of Valid/Invalid Responses on CMA-ES Attack: The CRPs (N_{CRP} = 4,000) used for training consist of a mixed combination of responses collected from the real PUF and fake PUF/RNG designs, depending on the percentage of fake responses. The proposed deception protocol is evaluated for two different challenge bit lengths, 64 and 128 bits.

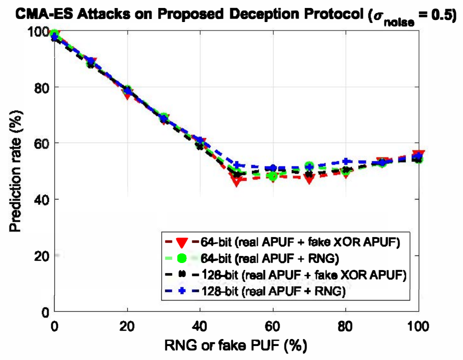

Figure 5. The CMA-ES attack results for the proposed deception protocol by applying different challenge bit lengths, 64-bit and 128-bit, as well as utilizing different fake response generators (RNG and XOR PUF). The number of training samples used for this experiment, N_{CRP} = 4,000, is the same as that used in [18].

Fig. 5 shows the CMA-ES attack results on the proposed deception protocol for different percentages of responses from the fake PUFs or RNGs with different challenge bit lengths. The prediction rates for both lengths of challenges decrease proportionally with x_f from 100% with all real responses to approximately 50% with 50% of responses generated from the fake PUFs/RNGs in the 4000 training samples. As the percentage of responses from the fake PUF or RNGs is greater than 50%, the correct prediction rate stays relatively constant and fluctuates around 50%, which is as good as wild guess. Moreover, there is no appreciable difference in the prediction rates of the CMA-ES attack by using either fake APUF or RNG to generate the fake responses for the proposed protocol. With more than 50% RNG or fake PUF used for the training, the prediction rates decrease to approximate 50% and fluctuate in a small range around 50%. The marginal (less than 5%) monotonic rise in prediction when near 100% of fake responses are used for training is due to overfitting. From the above experiments, the percentage of fake responses has a greater impact on the prediction rate of CMA-ES attack than LR attack.

B. *Effect of Dynamic and Static Waiting Time*

For every authentication attempt made by the adversary before the waiting time $\mathsf{T_w}$, the offset time $\mathsf{T_{os}}$ will be doubled.

Authorized licensed use limited to: Queens University Belfast. Downloaded on July 19,2023 at 14:01:38 UTC from IEEE Xplore. Restrictions apply.

Hence, the waiting time after applying N_f challenges with waiting time shorter than the current $\mathsf{T_w}$ is given by:

$$\begin{aligned}\mathsf{T_w} &= \left(\mathsf{P_{min}} + \mathsf{P_{os}} + 2\mathsf{P_{os}} + \cdots + 2^{N_f}\mathsf{P_{os}}\right)\mathsf{T_{clk}} \\ &= \left\{2^k + \mathsf{P_{os}}(2^{N_f} - 1)\right\}\mathsf{T_{clk}}\end{aligned} \tag{1}$$

For $\mathsf{T_{clk}} = 1ms$, $L = 32$, $k = 21$ and $\mathsf{P_{os}} = 1000$, $\mathsf{T_{min}}$ is less than an hour. After only $N_f = 25$, the waiting time $\mathsf{T_w}$ exceeds 388 days. If $\mathsf{P_{os}}$ is updated by the server to 2^{25}, it takes only $N_f = 10$ adversarial attempts for the waiting time $\mathsf{T_w}$ to exceed 397. The more the attacker attempts to close in their estimate of $\mathsf{T_w}$, the faster $\mathsf{T_w}$ will blow up to years. The attacker can derive a full set of valid training samples by always waiting for a time longer than $\mathsf{T_w}$ before sending the next challenge, provided that the genuine server did not authenticate the device to change the $\mathsf{P_{os}}$ during this period. If the attacker sends a challenge every t, where $t > \mathsf{T_w}$, the time, T, to obtain the training set can be calculated by (2), which includes the time to derive N_{CRP} of training data, the training time T_{train}, and the trial-and-error time $T_{t\&e}$ for determining $\mathsf{T_w}$.

$$T = N_{CRP} \times \mathsf{T_w} + T_{train} + T_{t\&e} \tag{2}$$

Omitting $T_{t\&e}$ and the penalty due to the dynamic component $\mathsf{T_{os}}$ (i.e., assume $\mathsf{T_{os}} = 0$), Fig. 6 shows that the efficiency of modeling attacks on existing unfortified ($\mathsf{T_w} = 0$ hr) 64-stage APUF reduces with different $\mathsf{T_{min}}$. Even if we assume the attacker knows $\mathsf{T_{min}}$, the overall attack time T without $\mathsf{T_{os}}$ can still be made impractically long by fixing $\mathsf{T_{min}}$ according to the known required number of training samples N_{CRP} for a reasonable accuracy of prediction for modelling the chosen genuine SPUF. For example, a 64-stage APUF design can be predicted with 95% accuracy in approximately 0.01 seconds (the time for sending and reading one CRP is neglected) with a training sample size of $N_{CRP} = 640$ using the above LR based experiment. Then using the proposed deception protocol with $\mathsf{T_{min}}$ of 1 hour, it will require 25.2 days for the same N_{CRP} to achieve the same prediction rate. Similarly, the overall attack time T is also longer when more training samples N_{CRP} are needed if a larger APUF with more number of stages or a more complex XOR-PUF is used. The original attack time increases to 0.13 seconds and the prediction rate hits 99% when N_{CRP} is increased to 2,555. By using the same static $\mathsf{T_{min}}$ of 1 hour with our protocol, without compromising the success rate of prediction, the overall attack time T will increase to 109 days. If $\mathsf{T_{min}}$ is set to 24 hours, the attacker will require approximately 7.1 years to achieve a prediction rate of 99%.

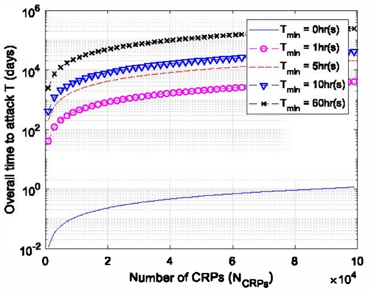

Figure 6. The overall time, T, taken by modeling attacks to predict a 64-bit APUF with respect to the minimum number of training samples N_{CRP} and static $\mathsf{T_{min}}$ of the proposed deception protocol. The typical training time, T_{train}, is assumed to be 1 second.

Table I
A Comparison of Proposed Deception Protocol Against Lightweight Protocol Finalists from [3] and Two Recent Published Protocols, Lockdown Protocol from [12] and Latent Obfuscation [13].

Protocol	PUF-independent	Modeling resistance	Controlled no. of auth.	TRNG
Basic authentication [2]	✓	✗	✓	✗
Slender PUF [9]	✗	✗	✗	✓
Noise bifur. [10]	✗	✗	✗	✓
Lockdown Ia [12]	✓	∽	✓	✗
Lockdown Ib [12]	✓	✓	✓	✗
Lockdown IIa [12]	✓	∽	✓	✓
Lockdown IIb [12]	✓	✓	✓	✓
Latent obfus. [13]	✗	✓	✓	✓
System-of-PUF [22]	✗	✗	✓	✗
Proposed protocol	✓	✓	✗	✗

✗ denotes 'no'. The symbol ✓ denotes 'yes'.
∽ denotes result not available.

IV. Protocol Comparison

The survey in [3] divides the PUF-based authentication protocols into two groups, *heavyweight* and *lightweight*. Most of the *heavyweight* protocols require either a strong cryptographic algorithm for privacy amplification and an ECC for response reconciliation [4]–[8]. For example, the *controlled PUF* [21] applies hashing to obfuscate its CRPs and requires ECC to correct the noisy responses. A detailed review and comparison of the *heavyweight* authentication protocols have already been done in [3]. In Table I, we compare the proposed deception protocol against the baseline, three other protocols, *slender* PUF [9], *noise bifurcation* [10] and *system-of-PUF* protocols [22] in the *lightweight* group, as well as two recently published authentication protocols, *lockdown* [12] and *latent obfuscation.* The evaluation metrics used in the comparison are:

- *PUF-independent* relates to whether or not the protocol requires a specific PUF design.
- *Modeling resistance* refers to the protocol assisted resistance to model building attacks.
- *Controlled no. of auth.* refers to a tight bound on quantity of CRPs that can be authenticated.
- *TRNG* refers to any TRNG component used in these protocols.

As analyzed earlier, the proposed deception protocol demonstrates good robustness to different ML based attacks. Any type of SPUF can be used as the genuine SPUF or fake SPUF. Among the previous works, only *lockdown* protocols have no restriction on the PUF design but its model-based authentication scheme is designed based on system-

Authorized licensed use limited to: Queens University Belfast. Downloaded on July 19,2023 at 14:01:38 UTC from IEEE Xplore. Restrictions apply.

level instantiation of hard-to-learn XOR-PUF. In contrast, our scheme enables secure model-based authentication with generic machine-learnable SPUF. Except *lockdown*, *latent obfuscation* and our proposed deception protocols, other authentication protocols listed in Table I are vulnerable to modeling attacks. Typically, a *TRNG* is used to generate a random substring to hide the response, *e.g.*, in slender PUF [9]. In our work, a temporal control is used to delay the attacker from collecting enough correct CRPs from the real SPUF within a practical time. Overall, the proposed deception protocol has achieved all the desirable properties. There is no tight control on the number of authentications except a minimum time limit is imposed between two legit challenges to an attacker who has no knowledge about the unused CRPs. This time limit is otherwise unrestrictive for the genuine server.

V. Conclusion

Deception as a defense strategy provides greater delay, confusion and disruption than rejecting sessions to the attacker's onslaught. It can drive preventive countermeasures to delay an attack, causing the adversary economic harm to figure out what is real and what is not, and hesitant to proceed. In this paper, we propose a novel deception authentication protocol by deceiving the adversary to use a training set dominated by fake/invalid responses for ML. This will prevent their PUF clone from correctly predicting the response to the unknown challenge. The rapid drop of prediction accuracy with increasing fraction of fake responses is demonstrated using two of the most widely known modeling attack techniques, LR and CMA-ES. The enrolled software model of real SPUF is used by the server to authenticate the device at any time through the proposed deception-based protocol. Each successful mutual authentication initiated by the server also randomly changes the dynamic component of the waiting time between two successive challenges, making it even more disruptive for the adversary to estimate the waiting time.

Acknowledgment

This work was partly supported by the Singapore MOE Tier 2 Grant No. MOE2015-T2-2-013, by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korean government(MSIT) (No. 2016-0-00399, Study on secure key hiding technology for IoT devices [KeyHAS Project]), by the Engineering and Physical Sciences Research Council (EPSRC) (EP/N508664/-CSIT2), by National Natural Science Foundation China (61771239) and by Nature Science Foundation of Jiangsu Province (BK20151477).

References

[1] C. H. Chang, Y. Zheng, and L. Zhang, "A retrospective and a look forward: Fifteen years of physical unclonable function advancement," *IEEE Circuits Syst. Mag.*, vol. 17, no. 3, pp. 32–62, Aug. 2017.

[2] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," *Sci.*, vol. 297, no. 5589, pp. 2026–2030, 2002.

[3] J. Delvaux, R. Peeters, D. Gu, and I. Verbauwhede, "A survey on lightweight entity authentication with strong PUFs," *ACM Comput. Surv.*, vol. 48, no. 2, pp. 26:1–26:42, Oct. 2015.

[4] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *Proc. 9th Int. Conf. on Comput. and Commun. Security (CCS'02)*. New York, NY, USA: ACM, 2002, pp. 148–160.

[5] S. Katzenbeisser, Ü. Koçabas, V. van der Leest, A.-R. Sadeghi, G.-J. Schrijen, H. Schröder, and C. Wachsmann, "Recyclable PUFs: Logically reconfigurable PUFs," in *Cryptographic Hardware and Embedded Syst. (CHES'11)*, B. Preneel and T. Takagi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 374–389.

[6] A. Van Herrewege, S. Katzenbeisser, R. Maes, R. Peeters, A.-R. Sadeghi, I. Verbauwhede, and C. Wachsmann, "Reverse fuzzy extractors: Enabling lightweight mutual authentication for PUF-enabled RFIDs," in *Financial Cryptography and Data Security*, A. D. Keromytis, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 374–389.

[7] Ü. Kocabaş, A. Peter, S. Katzenbeisser, and A.-R. Sadeghi, "Converse PUF-based authentication," in *Proc. Int. Conf. on Trust and Trustworthy Computing*. Springer, 2012, pp. 142–158.

[8] Y. Jin, W. Xin, H. Sun, and Z. Chen, "PUF-based RFID authentication protocol against secret key leakage," *Web Technologies and Applications*, pp. 318–329, 2012.

[9] M. Majzoobi, M. Rostami, F. Koushanfar, D. S. Wallach, and S. Devadas, "Slender PUF Protocol: A Lightweight, Robust, and Secure Authentication by Substring Matching," in *Proc. IEEE Symposium on Security and Privacy Workshops*, May 2012, pp. 33–44.

[10] M. D. Yu, D. M'Raihi, I. Verbauwhede, and S. Devadas, "A noise bifurcation architecture for linear additive physical functions," in *Proc. IEEE Int. Symp. on Hardware-Oriented Security and Trust (HOST'14)*, May 2014, pp. 124–129.

[11] U. Rührmair, C. Jaeger, M. Bator, M. Stutzmann, P. Lugli, and G. Csaba, "Applications of high-capacity crossbar memories in cryptography," *IEEE Trans. Nanotechnol.*, vol. 10, no. 3, pp. 489–498, May 2011.

[12] M.-D. Yu, M. Hiller, J. Delvaux, R. Sowell, S. Devadas, and I. Verbauwhede, "A lockdown technique to prevent machine learning on PUFs for lightweight authentication," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 2, no. 3, pp. 146–159, 2016.

[13] Y. Gao, S. F. Al-Sarawi, D. Abbott, A.-R. Sadeghi, and D. C. Ranasinghe, "Modeling Attack Resilient Reconfigurable Latent Obfuscation Technique for PUF based Lightweight Authentication," *arXiv preprint arXiv:1706.06232*, 2017.

[14] J. Delvaux, "Attacks on the puf-based authentication protocols yehl16 and gaomaar17," Cryptology ePrint Archive, Report 2017/1134, 2017, https://eprint.iacr.org/2017/1134.

[15] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proc. 17th Conf. on Comput. and Commun. Security (CCS'10)*. ACM, Oct. 2010, pp. 237–249.

[16] U. Rührmair, J. Sölter, F. Sehnke, X. Xu, A. Mahmoud, V. Stoyanova, G. Dror, J. Schmidhuber, W. Burleson, and S. Devadas, "PUF modeling attacks on simulated and silicon data," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 11, pp. 1876–1891, 2013.

[17] G. T. Becker, R. Kumar *et al.*, "Active and passive side-channel attacks on delay based PUF designs," *IACR Cryptology ePrint Archive*, vol. 2014, p. 287, 2014.

[18] G. T. Becker, "The gap between promise and reality: On the insecurity of xor arbiter PUFs," in *Cryptographic Hardware and Embedded Syst. (CHES'15)*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 535–555.

[19] ——, "On the pitfalls of using arbiter-PUFs as building blocks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 8, pp. 1295–1307, 2015.

[20] D. Lim, J. W. Lee, B. Gassend, G. E. Suh, M. Van Dijk, and S. Devadas, "Extracting secret keys from integrated circuits," *IEEE Trans. VLSI Syst.*, vol. 13, no. 10, pp. 1200–1205, 2005.

[21] B. Gassend, M. V. Dijk, D. Clarke, E. Torlak, S. Devadas, and P. Tuyls, "Controlled physical random functions and applications," *ACM Transactions on Information and System Security (TISSEC)*, vol. 10, no. 4, p. 3, 2008.

[22] S. C. Konigsmark, L. K. Hwang, D. Chen, and M. D. Wong, "System-of-PUFs: multilevel security for embedded systems," in *Proc. Int. Conf. on Hardware/Software Codesign and Syst. Synthesis (CODES+ISSS)*. IEEE, 2014, pp. 1–10.

Authorized licensed use limited to: Queens University Belfast. Downloaded on July 19,2023 at 14:01:38 UTC from IEEE Xplore. Restrictions apply.