Adaptive and Dynamic Device Authentication Using Lorenz Chaotic Systems

Lake Bu[†], Hai Cheng[§]*, Michel A. Kinsy[†], *Member, IEEE*
[§]Department of Electronic Engineering, Heilongjiang University, Harbin, China
[†]Adaptive and Secure Computing Systems Laboratory, Boston University, Boston, USA

***Abstract*—Chaotic systems such as Lorenz functions have been proposed as cryptographic primitives due to their short-range divergence attributes. They are commonly used in pseudo random number generators, key agreement protocols, and certain classes of encryption procedures. These functions are typically used for their chaotic behavior. However, two of their key properties are often overlooked: (1) their long-range convergence behavior is seldom used, and (2) the static nature of their system parameters is disregarded. The static nature of the system parameters, i.e., core secret, renders these functions vulnerable to a number of attacks when they are deployed in security applications. In this work, we examine these usage gaps and discover compelling security applications for these chaotic systems, in particular, Lorenz chaotic systems. In this paper, we propose an adaptive and dynamic authentication scheme based on discrete Lorenz chaotic systems. The scheme leverages Lorenz function's convergence to achieve a fast and lightweight authentication protocol. We also devise a dynamic parameter configuration technique to enhance the security of the protocol.**

Index terms — *Chaotic System, Lorenz Function, PUF, Authentication*.

I. Introduction

The Lorenz system, which consists of a set of differential equations, was originally proposed to describe and model thermally induced fluid convection in the atmosphere [1]. In a chaotic system like the Lorenz system, once a proper set of parameters is selected, the output of the chaotic functions is highly sensitive to the initial state. The slightest variation in the initial state values will result in a large change in the output values. Furthermore, given a number of outputs from a chaotic system, it is nearly impossible to precisely reverse engineer the parameters of the system. Consequently, predicting the system's future behavior is impracticable. However, because of its short-range divergence properties, it has found application in many other domains, the most common one being encryption. Researchers have suggested using these chaotic functions for block encryptions, image encryptions, digest generations, and key agreement protocols.

Besides its short-range divergence behavior, the Lorenz system also has a convergence property. This aspect of the system constrains all the outputs in a closed trajectory or map. Although the output (a point in the trajectory) is highly unpredictable due to the system's sensitivity to the initial state values, the trajectory's holistic shape and boundaries are fully determined once the system parameters are fixed. As a corollary, the chaotic system's parameters cannot be arbitrarily chosen or at random. They have to be selected and tested in a way that both the divergence and convergence properties are satisfied. Although the theoretical foundation of chaotic systems is well-established, their application thus far has been fairly narrow. For example in most applications such as obfuscation and encryption, only their divergence property is used, whereas their convergence attribute is seldom mentioned or exploited. Another central, often overlooked, feature of these chaotic systems is that their system parameters can be treated as the secret keys to unlock/predict both their short and long-range behaviors. Albeit, these keys are static and vulnerable to static secret key attacks such as key counterfeiting.

In this work, we broaden the application of chaotic systems, specifically Lorenz chaotic systems and develop an adaptive authentication protocol that is both efficient and secure. The major contributions of the work are:

1) an approach that takes advantage of both the divergence and convergence properties of the Lorenz systems in a way that it is hard for adversaries to predict, but easy for trusted parties to authenticate;
2) a technique to configure the system parameters dynamically using the intrinsic characteristics of the authenticated hardware device - this way, the scheme becomes much harder for adversaries to breach;
3) an authentication protocol that is adaptive and efficient - its runtime complexity and power are adjustable and proportional to the context of execution or application needs.

II. Preliminaries of the Lorenz Chaotic Systems and Physical Unclonable Functions

In this section we (i) introduce the key concept and properties of the Lorenz chaotic systems, and (ii) present the main notations from physical unclonable functions (PUFs) used for the dynamic parameter configuration in Section III. To better facilitate the presentation and understanding of the various points made in the paper, we adopt the following notations:

- α, β, γ: the system parameters of Lorenz functions;
- x, y, z: the outputs of chaotic functions;
- p_n: a point on the Lorenz function's trajectory/map. $p_n = (x_n, y_n, z_n)$;
- n and m: the dynamic and static numbers of iterations to run the Lorenz functions, respectively;
- $LF_i(p_0, n)$: a Lorenz function with system parameters $(\alpha_i, \beta_i, \gamma_i)$, and arguments of the initial state p_0 and number of iterations n;
- CHL_i: the i^{th} challenge to a PUF;
- $RSP_{j:i}$: the i^{th} response of a PUF indexed by j;
- CRP: the challenge and response pairs of a PUF.

A. Chaotic System by Lorenz Functions

A chaotic system is a type of nonlinear and unpredictable system which is highly sensitive to the initial conditions. In a such system, a slight difference in the initial state will produce rapid escalating and compounding variations in the system's future behavior. These phenomena are often described by

* Hai Cheng participated in this research while he was a visiting scholar with the Adaptive and Secure Computing Systems Laboratory.

fractal mathematics, which capture the infinite complexity of their nature. Important properties of chaotic systems are: initial condition sensitivity, unpredictability, fractals, divergence, and convergence. There are many types of chaotic systems. In this work we primarily focus on the Lorenz systems which are a 3D chaotic map. The discrete Lorenz chaotic functions are as follows:

$$\begin{aligned} x_{n+1} &= x_n + \alpha(x_n - y_n)\triangle t \\ y_{n+1} &= y_n + (\gamma x_n - x_n z_n - y_n)\triangle t, \\ z_{n+1} &= z_n + (x_n y_n - \beta z_n)\triangle t \end{aligned} \tag{1}$$

where (α, β, γ) are called the system parameters, and $\triangle t$ determines the resolution of the map. The parameters (α, β, γ) have to be carefully selected and tested to maintain the convergence of the Lorenz map. A statistical pattern of a Lorenz system is shown in Fig. 1. Unlike many random systems which only demonstrate divergence but not convergence, Lorenz systems have both properties.

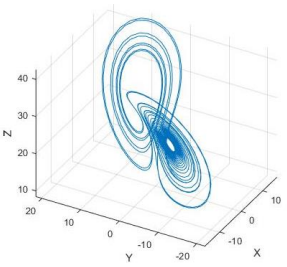


Fig. 1: The trajectory of a 3D Lorenz system, which usually has a butterfly pattern.

1) The divergence of Lorenz Systems: Intuitively, the divergence comes from the high randomness of the location and timing that a point $p_n = (x_n, y_n, z_n)$ appears on a 3D Lorenz map. Theoretically speaking, Lyapunov exponent can be used to measure the rate of divergence of a chaotic system:

$$|\delta(p)| \approx |\delta(0)|e^{\lambda p}, \tag{2}$$

where for a trajectory $T(p)$'s nearby orbit $T(p) + \delta(p)$, $\delta(p)$ is a vector with infinitesimal initial length. The maximal λ for Lorenz system is known to be approximately 0.9056 [2].

2) The Convergence of Lorenz Systems: Figuratively speaking, once a set of (α, β, γ) are given, the shape of the butterfly pattern is determined. In addition, even if the initial state p_0 is not a point on the trajectory, it will soon be attracted into the orbit within limited iterations. The convergence property can be described by Hausdorff dimension $dim_H K$ bounded by [3]:

$$dim_H K \leq 3 - \frac{2(\alpha+\beta+1)}{\alpha+1+\sqrt{(\alpha-1)^2+4\gamma\alpha}} \tag{3}$$

B. Physical Unclonable Functions (PUFs)

A physical unclonable function (PUF) is a piece of hardware that will produce an unpredictable response to a challenge due to its manufacturing variations. Each response is an output of a nonlinear function using the stimulus (challenge) and the PUF's own unique physical properties - "silicon fingerprints". Even with the same circuit layout and manufacturing process, two pieces of hardware will still have distinct behaviors under the same challenge [4]. PUFs are mostly used to verify the validity of a hardware device. In this work, it is serving a slightly different purpose: dynamic parameter updating. The procedure works as follows:

(i) Before a PUF device, indexed by j, is deployed, the server or verifier uses $\{CHL_0, CHL_1, \cdots, CHL_i, \cdots\}$ as the challenges to the PUF. Then the corresponding responses $\{RSP_{j:0}, RSP_{j:1}, \cdots, RSP_{j:i}, \cdots\}$ are stored;
(ii) When the server chooses to use $RSP_{j:i}$ to generate the new parameters on both *itself and the device*, it needs to inform the device of the choice. To do so, the server sends CHL_i to the device;
(iii) The PUF applies CHL_i to locally retrieve the response $RSP_{j:i}$, which is used to generate the new parameters.

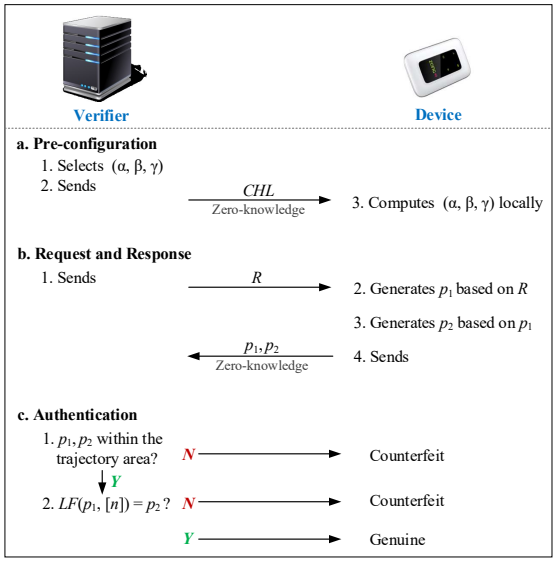
The above procedure is secure against eavesdropping since CHL_i leaks zero knowledge of $RSP_{j:i}$.

III. Adaptive Authentication Using Lorenz Chaotic Systems

We first give an overview of the protocol followed by the detailed description of each step of the protocol in the subsections. The advantages of the proposed protocol are: (1) it is hard for an adversary to predict outputs of the system due to the Lorenz functions' divergence behavior, but it is easy for a verifier to authenticate these outputs because given a specific set of system parameters, the global behavior is deterministic; (2) it fixes the integer digits of the Lorenz system parameters to guarantee the convergence of the system while keeping the decimal digits reconfigurable in order to dynamically control the short-range divergence of the system - this technique enhances the security of the protocol by introducing controllable variability to the system parameters; and (3) the authentication is performed in an adaptive manner for algorithmic efficiency.

The proposed adaptive authentication protocol is:

Protocol III.1.



In the configuration step, the verifier dynamically determines the Lorenz system parameters. Then, it shares them with the device in a zero-knowledge way. At the time of authentication, the verifier sends a random number R as the request to the device. The device responds with two points $\{p_1, p_2\}$ on the trajectory/map while leaking no knowledge of the system parameters. To verify the authenticity of the device, the verifier first examines if $\{p_1, p_2\}$ are both within the Lorenz map's pattern boundaries, which is a fast but coarse-grained filter. If they are, the verifier then checks if p_2 can be computed by the Lorenz functions with p_1 being the initial condition.

This step is fine-grained and accurate. If either verification step fails, the device is identified as a counterfeit. Most counterfeits will be detected at the coarse-grained verification. This two-step authentication approach saves compute resources and execution time by performing the fine-grained stage of the authentication only for a higher resolution validation.

A. *Step 1: Pre-configuration - Dynamic Parameter Updating*

To mitigate some of the security vulnerabilities associated with the use of static system parameters, we proposed an approach to dynamically configure the parameters, while still keeping the chaotic maps convergent. A hardware primitive named PUF-seeded Lorenz functions (PSL, Fig. 2) is implemented on each device to enable this functionality.

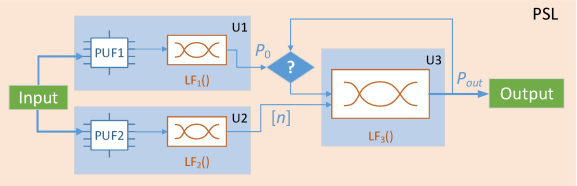


Fig. 2: The system parameters of the Lorenz functions in U1, U2, U3 are $(\alpha_1, \beta_1, \gamma_1)$, $(\alpha_2, \beta_2, \gamma_2)$, and $(\alpha_3, \beta_3, \gamma_3)$, respectively.

U1 and U2 serve as the parameter updating units; they generate the system parameters for U3. The functions $LF_1()$ and $LF_2()$ are randomness amplifiers for PUF1 and PUF2 [5]. Their input response parameter and iteration number m are set statically in the same manner as the parameter selection for conventional Lorenz systems. It is worth noting that a single PUF module could be used for dynamic configuration. In our illustrative case, two PUFs are used to extend the collective range. The final system parameters for the PSL are $(\alpha_3, \beta_3, \gamma_3)$. In this work, we use a 64-bit vector fixed-point representation for the three system parameters. It is also the maximum data width supported by our FPGA IP multiplier. The first 8 bits represent the integer part and the last 56 bits the decimal portion. The 56 decimal bits are further partitioned into 8 bits and 48 bits for parameter reconfiguration purposes. Other bit configurations can be used as well. Generally, Lorenz systems parameters tend to range from 1 to less than 200. Therefore, an 8-bit width template is sufficient to represent their integer parts.

Protocol III.2. The dynamic configuration of the lower 48 decimal bits of $(\alpha_3, \beta_3, \gamma_3)$ is performed in following manner:

1) The verifier acquires k pairs of CRPs from both PUF1 and PUF2. The 8-bit integer part and the first 8 decimal bits of the parameters $(\alpha_3, \beta_3, \gamma_3)$ are always fixed and stored as (a, b, c), in order to maintain the system convergence;
2) When the last 48 decimal bits of $(\alpha_3, \beta_3, \gamma_3)$ are configured, the verifier selects three arbitrary responses $RSP_{1:i_0}$, $RSP_{2:i_1}$, $RSP_{2:i_2}$ from PUF1 and PUF2 (the last two responses target PUF2 in this case). Then, the verifier computes:
$$\begin{aligned} a_d &= LF_1(RSP_{1:i_0}, m) \\ b_d &= LF_2(RSP_{2:i_1}, m) \\ c_d &= LF_2(RSP_{2:i_2}, m) \end{aligned} \tag{4}$$
3) The parameters $(\alpha_3, \beta_3, \gamma_3)$ for this round are dynamically computed as:
$$\alpha_3 = (a||a_d), \quad \beta_3 = (b||b_d), \quad \gamma_3 = (c||c_d), \tag{5}$$
where $||$ is the concatenation operator.
4) To inform a device of the newly computed system parameters, the verifier sends challenges $CHL_{i_0}, CHL_{i_1}, CHL_{i_2}$ to the PSL. The system parameters are then locally regenerated on PSL side using the challenges.

Remark III.1. As is often the case with PUF implementations, the raw responses to different challenges have a very limited Hamming distance due to the increasing precision in fabrication process. Therefore, the responses generally need to be further randomized. In our protocol, the functions $LF_1(), LF_2()$ are added for this purpose. Fig. 3 shows how the Hamming distance between two 16-bit PUF responses are magnified. Another key benefit of U1 and U2 in the architecture is that PUF1 and PUF2 can very small and still provide a large pool of random vectors.

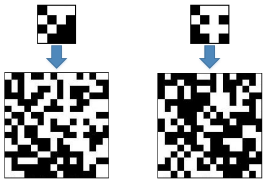


Fig. 3: The two original 16-bit responses (upper left and right) have only a 4-bit difference. After being randomized by Lorenz functions, they become 256-bit responses with a 131-bit difference.

B. *Step 2: Request and Response*

In this step, the verifier sends a random number R to the PSL as the request. This number is used as the challenge for both PUF1 and PUF2. The module U1 generates p_0 - the initial point (x_0, y_0, z_0) and U2 produces the dynamic iteration number n. The function $LF_3()$ takes in p_0 and n as its initial condition inputs (cf. [Eq. 6]). It should be pointed out that p_0 does not have to be in the trajectory of $LF_3()$. In any case, the function $LF_3()$ will quickly be drawn to its prescribed path. The module U3 generates the first response by:

$$p_1 = LF_3(p_0, [n]). \tag{6}$$

Next, p_1 is used to generate the second response:

$$p_2 = LF_3(p_1, [n]), \tag{7}$$

where $[n]$ denotes the lower and upper bounds of the dynamic iteration number n. The lower and upper bounds are symbolized as u and v, respectively, and expressed in the form:

$$[n] = n \bmod (v - u) + u. \tag{8}$$

Both $\{p_1, p_2\}$ are then sent back to the verifier. Since R is newly generated for every authentication to make the corresponding $\{p_1, p_2\}$ unpredictable, $(\alpha_3, \beta_3, \gamma_3)$ do not need to be reconfigured very often. It is only necessary when the verifier believes there is a leakage of the parameters. We omit the detailed analysis of the parameters' information leakage for brevity.

Remark III.2. It should be underlined that there is a minimum value for n to ensure that p_1 lands on the trajectory from an arbitrary point p_0. This number depends on the resolution variable $\triangle t$ in [Eq. 1] as shown in figure 4. In our illustrative design case of 64-bit parameter values with 48 reconfiguration bits, we have $\lfloor n \rfloor \leq 217$. The upper bound is simply a function of the acceptable computational complexity for the verifier. In this work, the upper bound is set to 300, i.e., $217 \leq [n] \leq 300$.

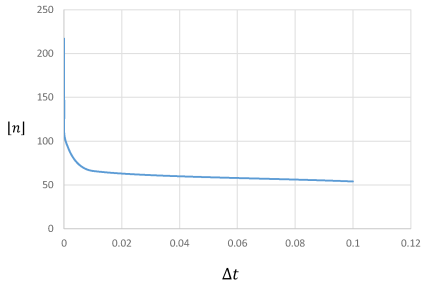


Fig. 4: The larger $\triangle t$ is, the less iterations are needed for an arbitrary point to be attracted to the prescribed trajectory.

C. Step 3: Adaptive Authentication

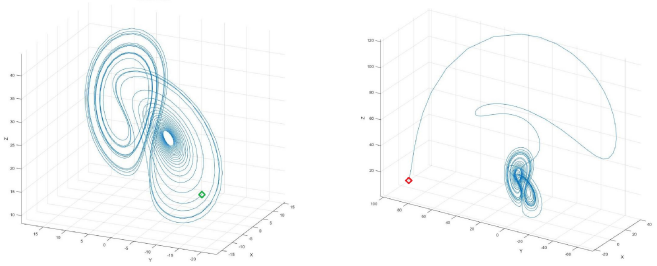
The adaptive authentication has two phases: coarse-grained and fine-grained verifications. The former is capable of detecting most counterfeit devices with minimal compute effort. The latter aims at recognizing more sophisticated forgeries that can guess, with a high degree of accuracy, the parameters α_3, β_3, and γ_3 of the U3 module.

1) Coarse-grained Verification: This part of the protocol takes advantage of the convergence property of Lorenz systems. The key observations are that (i) all the points generated by $LF_3()$ will be on the predetermined Lorenz map, (ii) this map is bounded, and (iii) two or more points generated with a different set of system parameters have a very high probability of falling outside the map. More importantly, the dimensions of the trajectory can be conveniently estimated by taking a relatively large $\triangle t$ in [Eq. 1], say $\triangle t = 0.01$. This way a low resolution Lorenz map can be drawn quickly with only a small number of points. Although this map is rather rough, it does provide an approximate bound for (x, y, z).

The bound is expressed as $\{[\bar{x}, \underline{x}], [\bar{y}, \underline{y}], [\bar{z}, \underline{z}]\}$, where $\bar{}$ denotes the maximum value of the coordinate and $\_$ the minimum value. The following equation checks the authenticity of a point:

$$\textbf{Valid} = (x \in [\bar{x}, \underline{x}] \ \& \ y \in [\bar{y}, \underline{y}] \ \& \ z \in [\bar{z}, \underline{z}])?1 : 0. \quad (9)$$

Fig. 5 shows an example of such a verification process. For the valid green point in (a), its coordinates are within the butterfly's area bounded by [Eq. 9]. As for the invalid red point in (b), its coordinates are both out-of-bounds and off the system trajectory. Therefore, its verification process will consist of (a) first pulling the point into the system's bounded map, and then (b) checking that it is also on the proper trajectory.



(a) On trajectory (b) Off trajectory

Fig. 5: The trajectory in both figures has the same parameters.

2) Fine-grained Authentication: In cases where an adversary makes a good estimate on the parameters α_3, β_3, and γ_3, they can generate a similar butterfly pattern whose area may overlap with the original pattern, as shown in figure 6. In such cases, the coarse-grained verification may not be highly accurate and can lead to some false positives. Therefore, we propose the invocation of the fine-grained verification procedure of the protocol:

$$p_2 \stackrel{?}{=} LF_3(p_1, [n]). \quad (10)$$

If the points p_1, p_2 are both on the trajectory, the equality in [Eq. 10] will hold. Otherwise it will fail. Figure 7 shows a case each for passing and failing the fine-grained authentication step.

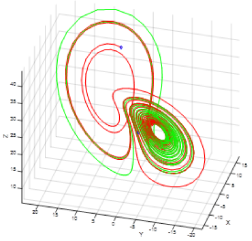
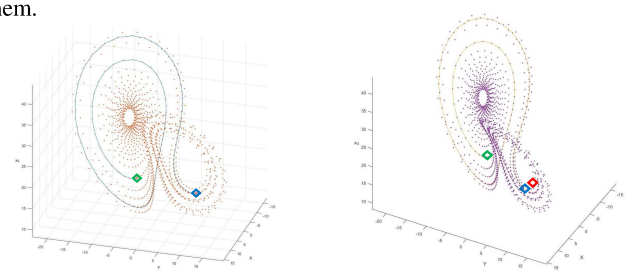


Fig. 6: The red and green trajectories are generated by parameters with a 10^{-5} difference, and there is certain area overlap between them.



(a) (b)

Fig. 7: (a) shows after $[n]$ iterations, p_2 (blue) is successfully computed with p_1 (green) being the initial condition. In (b), the actual p_2 (red) turns out to be an invalid point different from $LF_3(p_1, [n])$ (blue).

IV. Conclusion

In this paper, we introduce an adaptive authentication scheme based on Lorenz chaotic functions. It leverages both the divergence and convergence properties of the Lorenz systems to provide a secure and efficient device authentication approach. In addition, to mitigate some of attack vulnerabilities associated with static system parameters, the proposed scheme manages to update the parameters dynamically.

V. Acknowledgments

This research is partially supported by the NSF grant (No. CNS-1745808).

References

[1] F. C. Moon, *Chaotic and fractal dynamics: introduction for applied scientists and engineers*. John Wiley & Sons, 2008.

[2] D. Viswanath, "Lyapunov exponents from random fibonacci sequences to the lorenz equations," *Cornell University*, 1998.

[3] A. Y. Pogromsky, G. Santoboni, and H. Nijmeijer, "An ultimate bound on the trajectories of the lorenz system and its applications," *Nonlinearity*, 2003.

[4] L. Bu and M. A. Kinsy, "Weighted group decision making using multi-identity physical unclonable functions," *The International Conference on Field-Programmable Logic and Applications (FPL)*, 2018.

[5] L. Lin, H. Huang, and S. H, "Lorenz chaotic system based carbon nanotube physical unclonable functions," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2017.