Resonance Energy Transfer Based Physical Unclonable Function

by

Likun (Sam) Xi

Department of Electrical and Computer Engineering

Duke University

B.S.E. Thesis

2013

Abstract

Physical unclonable functions (PUFs) have been proposed in the past as a method of hardware token based authentication, but because PUFs cannot be duplicated, they cannot be used as secret keys in an encrypted communication session. Here, we demonstrate a novel PUF built on a self-assembled DNA lattice that functions using resonance energy transfer (RET). This PUF, which we call a RET-PUF, possesses a trapdoor property that allows it to be cloned if its exact structure and composition is known; its structure is too small for any attacker to discern by any currently known technology. We describe a cryptographic system using the unique properties of the RET-PUF. Theory shows that the RET-PUF cryptosystem is secure against a variety of applicable attacks and experiment verifies the correctness of the system. By demonstrating the strength and performance of the RET-PUF against existing cryptographic systems, we conclude that the RET-PUF is a viable shared secret key for secure communication.

Contents

Acknowledgements

I would like to thank Professor Chris Dwyer for his enduring mentorship and support over the past two and a half years. This project has been a long journey and I could not have gotten here had he not stuck with me the whole way, giving me lots of advice and exposing myself to many different areas in the field. Also, I want to thank Vishwa Nellore for partnering with me on this endeavor. She's done so much work on this project and I've learned a great deal about research and science from her. The rest of the lab: Alvy, Craig, Heather, Arjun, Mohammad, Siyang, Jun, and Viresh have been fantastic to me as well and I will miss all of them next year.

This thesis is dedicated to my family. My parents have worked hard to financially support my college education and give me the motivation and drive to pursue a career in research. My younger sister, now in high school, always gives me reason to come home more often, and I couldn't be more proud to have her as my sibling. I truly appreciate all the opportunities my family has created to let me pursue an education, build my career, and live life to its fullest. I love you all!

1

Overview

1.1 One-Way functions

The security of cryptographic protocols stems from an asymmetry in the amount of work required for a legitimate user to correctly decrypt an encrypted message as compared to an attacker. Mathematical one-way functions form the basis of many modern day cryptosystems. One-way functions are functions that are easily evaluated for any input, but difficult to invert given any output. Informally, a function is considered one-way if no efficient probabilistic algorithm can invert the function with greater than negligible probability. Although one-way functions have not actually been proven to exist, in practice, their existence is conjectured because no efficient algorithm to invert them has yet been found. RSA, one of the most commonly used cryptographic systems, uses modular exponentiation as the encoding mechanism and relies on the difficulty of factoring a very large number for its security [14]. Factoring large numbers is one conjectured one-way function, in that a product of two numbers can be easily computed, but the factors of that product cannot be easily determined. One-way functions that are used in cryptosystems are known as *trapdoor* one-way functions because they become trivial to invert if some secret information is known. This secret information is contained in a key that is kept private. In the case of RSA, factoring a product of two large primes is simple if one of the primes is known, so the trapdoor (and thus the keys used in the system) for the factoring problem is knowledge of one factor. The trapdoor property is required for such applications, because without it, the ciphertext becomes as indecipherable to a legitimate user as to an attacker.

1.2 Physical Unclonable Functions

Physical unclonable functions (PUFs) are physical manifestations of mathematical one-way functions [11]. The input is a physical stimulus that elicits a response from the PUF. Like their mathematical counterparts, PUFs need to be easy to evaluate but hard to invert, and their output must be highly unpredictable and change considerably in response to a small perturbation of the input. Responses should be unique to both the PUF and the input stimulus. A strong PUF has the quality that its output reveals no information about the PUF's internal structure and be relatively cheap and simple to produce en masse but computationally infeasible to model accurately. Some existing PUFs rely on manufacturing defects that are inherently random, which makes them virtually impossible to precisely duplicate; furthermore, their internal complexity makes them computationally infeasible to accurately model, which makes them a perfect candidate for authentication through the challenge-response pair framework.

Challenge-response pair authentication using PUFs goes as follows. A challenge in the form of an input stimulus or physical probe is applied to the PUF, and the PUF responds with a corresponding signature that is unpredictable and reveals no information about the PUF's internal structure. Two PUF models often drawn as examples are the speckle pattern PUF [11] and the CMOS delay PUF [19]. The speckle pattern PUF consists of a collection of dark particles, randomly distributed through a mixing process, embedded in a transparent medium. A challenge in the form of a coherent light beam is applied, giving rise to a speckle pattern that is encoded in a binary format and used as a key or unique identifier. Such a PUF is highly sensitive to the angle of the incident light. CMOS delay PUFs use two copies of the same digital circuit designed with race conditions. The input to such a PUF is a set of binary signals, and the circuit that completes before the other sets a bit in the output. Inherent variations in manufacturing results in every copy of this circuit responding differently to the same input. The final state or output of the circuit is a signature representing the PUF's response.

1.3 Resonance Energy Transfer

Resonance energy transfer (also known as Förster resonance energy transfer or FRET) is a radiationless transfer of energy between a donor and an acceptor fluorophore. A fluorophore is a molecule that enters an excited electronic state when it absorbs a photon and then releases a photon at a lower energy state when the fluorophore relaxes into its ground state. When a donor fluorophore absorbs a photon and enters its excited state, if an acceptor fluorophore is in close proximity, the donor photon can transfer its energy to the acceptor fluorophore through various nonradiative means (that is, de-excitation without photon fluorescence). The transferred energy

is sometimes called an *exciton*, representing a discrete packet of energy that is passed between fluorophores. The acceptor fluorophore absorbs the exciton, enters its excited state, and then relaxes into its ground state after some time. There are several de-excitation pathways for this acceptor fluorophore. It could potentially initiate RET to another fluorophore, or it could merely release a photon. There are other de-excitation pathways as well [21].

The efficiency of RET, also known as the fluorescence quantum yield, is the fraction of excited photons that undergo de-excitation to their ground states with fluorescence. It can be expressed as the ratio of rate constants for the various de-excitation pathways of this system. These rate constants are dependent on the type and structure of the fluorophores, the amount of overlap between the emission spectrum of the donor and the excitation spectrum of the acceptor, the distance between them, and their relative dipole moment orientations. The efficiency is given by [21]:

$$\phi_F = \frac{k_T}{k_T + k_D} \tag{1.1}$$

where k_T is the rate constant for transfer between donor and acceptor, and $k_D \propto 1/\tau_D$, where τ_D is the excited state lifetime of the donor fluorophore in the absence of transfer. Förster's derivation of k_T allows us to relate the fluorescence quantum yield to physical quantities:

$$\phi_F = \frac{1}{1 + (r/R_0)^6} \tag{1.2}$$

where R_0 is the Förster radius, defined as the distance between donor and acceptor at which spontaneous fluorescence of the donor or RET are equally probable, and r is the physical distance between donor and acceptor. R_0 is a function of the amount of overlap between the emission spectrum of the donor and the excitation spectrum of the acceptor (equation not shown). The fluorescence of a single fluorophore, which we define as its "impulse response", is described by a decaying exponential function $I_F(t) = k_r e^{-t/\tau_s}$ where τ_s is the excited state lifetime of the fluorophore and k_r is the rate constant of radiative de-excitation with fluorescence [21]. In a RET network, where multiple fluorophores may fluoresce, the fluorescence can be described by a multi-exponential curve $I_F(t) = \sum_i^n k_r^i e^{-t/\tau_i}$, where each τ_i is a function of the excited state lifetimes of the fluorophores that comprise the RET network and k_r^i is the corresponding rate constant of radiative de-excitation. These are conceptual simplifications; there are many possible conditions that change the overall structure of a RET network's fluorescence, like saturation, photobleaching, etc. In general, the response to an input light stimulus of a RET network can be characterized by the relative amplitude and lifetimes of the exponential decay functions that comprise the overall decay curve. Decomposition of multi-exponential decay functions have been studied extensively [3, 22]; the general consensus is that this is a very difficult problem.

Additional complexity of the response must be considered due to saturation of the network. A fluorophore is said to be saturated when it is unable absorb more photons or excitons because it is already in its excited state. Saturation effectively removes fluorophores from the interaction graph because that fluorophore cannot participate in RET until it has relaxed via fluorescence, RET, or other de-excitation pathways. Therefore, if we cascade multiple light pulses in rapid succession such that the time interval between pulses is shorter than the excited state lifetimes of the donor fluorophores, those fluorophores may not relax back to their ground states before the next photon arrives, and thus any subsequent fluorescence events due to subsequent stimuli are now the result of a *different* interaction graph with different transfer rates, Förster radius, and so on.

In addition to saturation, fluorophores suffer from an effect called photobleaching, in which the emission intensity of a fluorophore decreases over time as it is excited more and more until its contribution to the RET interaction graph is negligible. This is due to chemical and molecular bonding changes induced by excitation of the fluorophore to higher energy states. The precise mechanisms of photobleaching, however, are poorly understood. It is clear, however, that photobleaching attenuates the intensity of a fluorophore's fluorescence and changes the ability of a fluorophore to nonradiatively transfer energy to another fluorophore [21]. As we will see, photobleaching plays a major role in the properties of the RET-PUF cryptosystem.

RET is a thermally stable process. It is often modeled as an emission and immediate absorption of a "virtual" photon - virtual because its existence would violate the law of conservation of energy. Since this photon does not actually exist, it cannot be affected by ambient temperature. Therefore, we do not have to guarantee the exact same operating conditions in order to ensure reliability. RET is more greatly affected by the viscosity of the solvent, which determines the time scale of diffusion of fluorophores within the medium, and the fluorophores' geometric placement [21]. By embedding the fluorophores onto a DNA lattice, we eliminate all translational degrees of freedom and thus eliminate diffusion as a possible source of error.

For further details on resonance energy transfer and molecular fluorescence, we refer the reader to Bernard Valeur [21].

2

The RET-PUF Cryptosystem

2.1 The RET-PUF

The RET-PUF consists of fluorophores bound to a self-assembled DNA lattice. We use standard methods of DNA self-assembly to construct the lattice hierarchically and conjugate fluorophores to specific base pairs. Conceptually, to form the lattice, we first construct *tiles*, which are the intersection locations of the grid. The arms of the tiles are specially crafted sticky ends that will only bind to one other arm of another tile. Completed tiles are then mixed with other tiles to form the entire grid. Fluorophores are conjugated to specific base pairs on the grid. If the fluorophores are placed close enough to each other, they form a network in which resonance energy transfer can occur[1]. The RET-PUF behaves similarly to previously described PUFs in many ways. When a sequence of laser pulses is applied to the RET-PUF, it responds to the excitation via RET, which triggers fluorescence from the RET-PUF. The characteristics of this response are highly dependent on the wavelengths of the input pulses, because the wavelength of the excitation pulse selectively activates certain fluorophores and trigger RET in certain regions of the interaction graph. The relative position of the fluorophores determines the Förster radii between fluorophore pairs. The Förster radius, which has an inverse sixth power dependence on the inter-fluorophore distance, plays a large role in the efficiency of RET.

The self-assembly process enables us to attach fluorophores to any nucleotide base pair on the lattice. This level of fine-grained control over the location of the fluorophores is capable of defeating any current super-resolution microscopy technique, as in theory, we could bind fluorophores as close as 0.34 nm, the separation distance

[1] For further details about the DNA self-assembly process, we refer the reader to the Supplementary Materials of [13].

between adjacent base pairs. As we will show, an outside party who has unfettered physical access to the RET-PUF cannot decipher its structure with enough accuracy to replicate the RET-PUF accurately. However, the manufacturer of the RET-PUF is capable of making many identical copies of the structure because only the manufacturer is privy to the exact structure and composition of the RET-PUF. This information forms the cryptographic trapdoor, a critical property of the RET-PUF that, to the best of our knowledge, no other PUF possesses. The trapdoor enables the RET-PUF to be distributed to separate parties and act as an private key in a secure communications protocol as well as an authentication token.

The security of our cryptosystem relies on the following assumptions, which will be justified over the course of this thesis.

- The response of the RET-PUF is highly sensitive to its structure and the stimulus applied.
- A small change in the structure and/or composition of the RET-PUF or the applied stimulus will trigger an unpredictable but consistent change in the response of the RET-PUF.
- Any physical tampering of the RET-PUF can be detected while simultaneously rendering the RET-PUF useless.
- Knowledge of which fluorophores are on the RET-PUF is insufficient information to replicate or predict the RET-PUF's behavior.
- The RET-PUF's response characteristics evolve as it is used.
- The RET-PUF is a private key which automatically expires after a certain number of uses.

2.2 Cryptosystem Terminology

Fundamentally, the cryptographic system is a symmetric-key block substitution cipher, but the RET-PUF makes this system incomparably stronger than any traditional substitution cipher. Before we detail each of the major steps in the encryption and decryption functions, we define some key terms.

- Excitation: a sequence of laser pulses at specific delays and wavelengths. Excitations map directly to a block of plaintext bits. The number of laser pulses that comprise the excitation is variable, but it depends on the radix of the symbol space to be encrypted.
- Initial condition: a sequence of laser pulses at specific delays and wavelengths. Initial conditions are used to establish a common state of the RET-PUF before each block of data is encrypted. There usually doesn't need to be more than

two pulses. Initial conditions help two identical RET-PUFs in possibly different environments respond similarly and consistently.

- IX block: a set of initial conditions followed by a series of excitation pulses. Note that every IX block in a communication session will use the same the initial conditions.
- IX sequence: a sequence of IX blocks. IX sequences are mapped to a block of plaintext bytes. This will be described in greater detail later.
- Observation time window: a window of time after the excitation pulses to record the fluorescence behavior of the RET-PUF.
- Observation wavelength: the wavelength at which fluorescence is to be observed. Since the RET-PUF is comprised of different fluorophores that fluoresce at different wavelengths, one wavelength must be selected for observation.
- IX partition ID: an identifier for a partition of the complete IX space.
- Nonce: a large random value that is applied to the message before encryption. Nonces, which are only used once per message, increase entropy in the ciphertext bitstream and defend against a variety of attacks.

2.3 Cryptosystem

2.3.1 Definitions

We will use the following notation and conventions in this thesis.

1. An input to a RET-PUF is modeled as an impulse train $x(t, \lambda) \in \mathcal{X}$, in which each impulse may be at a different wavelength and $\mathcal{X}$ is the set of all finite length impulse train functions that represent inputs to a RET-PUF.
2. $r(t, \lambda) \in \mathcal{R}$ is a single response to a single input $x(t, \lambda)$, and $\mathcal{R}$ is the set of all multi-exponential decay curves that represent responses of a RET-PUF.
3. $\mathcal{X}_i$ is a partition of the complete input space $\mathcal{X}$ with IX partition ID i.
4. A RET-PUF is a transformation $\mathrm{P} : \mathcal{X} \rightarrow \mathcal{R}$.
5. $\mathrm{D} : M \rightarrow \mathcal{X}_i$ is an invertible transformation that maps a message block to a set of inputs $x(t, \lambda)$ in the partition $\mathcal{X}_i$ and vice versa. It is colloquially known as the *mapping database*.
6. $\mathrm{N}(r(t, \lambda))$ is a function that computes a cryptographic nonce from a RET-PUF response.

2.3.2 Assumptions

- The mapping database D is public knowledge and does not change.
- The following information specific to a communication session is sent in plaintext to the recipient:
  - The IX block $x_n(t, \lambda)$ used to generate a nonce from the RET-PUF.
  - An observation wavelength λ_{obs}.
  - An observation time window (t_{start}, t_{end}).
  - i, the partition of $\mathcal{X}$ used in this session.

2.3.3 Encryption

To encrypt a message M using a RET-PUF P, proceed as follows.

Represent M as a series of bytes. For long messages, divide M into smaller blocks and repeat this procedure for each message block.

Randomly select an IX block $x_n(t, \lambda)$ from $\mathcal{X}_i$, apply it to P, then evaluate N on the output to get the nonce n. Apply n to M to produce $M' = M \oplus n$. n can be used for multiple blocks. Formally, n is given by:

$$n = \mathrm{N}(\mathrm{P}\{x(t, \lambda)\}) \tag{2.1}$$

D maps a message M to a set of inputs $\{x_1, x_2, ..., x_n\}$. Randomly select an $x(t, \lambda)$ from $\mathrm{D}\{M'\}$. Apply x on P to get $r(t, \lambda) = \mathrm{P}\{x(t, \lambda)\}$. Now time window r on the interval (t_{start}, t_{end}) and an observation wavelength $\lambda = \lambda_{obs}$ to obtain the ciphertext c, which is therefore given by:

$$c(t) = r(t, \lambda_{obs})[u_t(t_{start}) - u_t(t_{end})] \tag{2.2}$$

2.3.4 Decryption

To decrypt a ciphertext $c(t)$, proceed as follows.

Apply $x_n(t, \lambda)$ to P to get $r_n(t, \lambda)$ and compute $n = \mathrm{N}(r_n(t, \lambda))$.

Now, iterate through all IX blocks in $\mathcal{X}_i$ and identify the input x that produces a response $r(t, \lambda)$ which best matches $c(t)$ at $\lambda = \lambda_{obs}$ according to a ranking metric, such as one that minimizes a "distance" between r and c.

Evaluate $M' = \mathrm{D}^{-1}(x)$ and recover the original message $M = M' \oplus n$.

2.3.5 Nonce

To generate a nonce for a communication session, proceed as follows.

Choose an IX block $x_n(t, \lambda)$ and apply it to P to get $r_n(t, \lambda)$. Now, take the standard Hough transform of $\ln(r_n)$. The parameters to the Hough transform are flexible; in our experiments, we used a ρ spacing of 10 and θ in increments of 0.2°.

Treat the plot of the Hough transform as a bitmap. From this bitmap, extract a set of bits from a row. These bits form the nonce. To enable the receiving party to use this same set of bits, apply a Reed-Soloman error correcting code to this set of bits. Send the error correcting code segment to the recipient. We leave how to determine the ideal row and bits from this bitmap for future work, as it requires detailed characterization of the Hough space.

3

Methods

3.1 TREX

We devised an optical apparatus, called TREX, that enables us to perform time-resolved measurements of a RET-PUF's excitation spectrum. Figure 3.1 shows the setup in detail. Briefly, we split a single coherent laser beam of white light into four separate beams via a series of beam splitters and mirrors (but we could have more, depending on how many input channels we need). On each channel, narrow laser line filters on the outputs of the beam splitters select a wavelength for an input channel. These filtered beams travel down variable length tracks, are turned around with Dove prisms at the end of their tracks, and then are deflected with right angle prisms onto the cuvette containing the sample. The variable length tracks enable us to precisely choose the time at which each laser beam strikes the RET-PUF. Another laser line filter is applied on the output port of the cuvette holder. The photons that pass through this output filter are focused onto the aperture of a single photon avalanche detector (SPAD), which sends its data to a time-correlated single photon counter (TCSPC) module attached to a computer.

The Dove prisms are mounted on two translation stages attached to an optical post and a rail carrier, which moves back and forth on a dovetail rail bolted on an optical breadboard. One translation stage allows us to laterally align the Dove prisms with the laser beam and the right-angle prism that redirects the beam onto the cuvette containing the sample. The other translation stage lets us adjust the precise distance that the beam must travel before being turned around. The translation stages have a 12.7mm travel range, and one complete revolution of the adjustment knob on the stage moves the stage 0.35mm [6], which translates into about 6.6ps of additional delay on that particular channel. Therefore, we can obtain precise delay times by

adjusting the rough position of the post and then fine tuning through the translation stages.

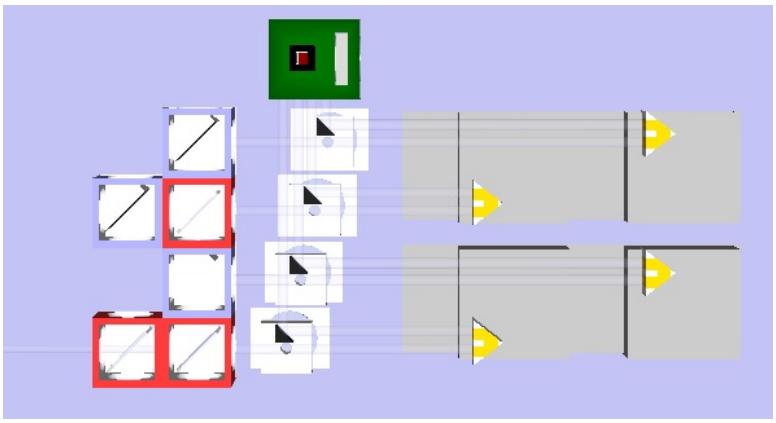


FIGURE 3.1: Structure of TREX. SPAD is not shown.

We use the SuperK Extreme class IV supercontinuum white light laser as our excitation source. The laser produces coherent light over the wavelength range 400nm to 2400nm, from 1.5W to 8W total power. We operate the laser at a repetition rate of 80MHz, where each pulse has a FWHM of 0.7ns [12]. Figure 3.1 shows the power density of this laser.

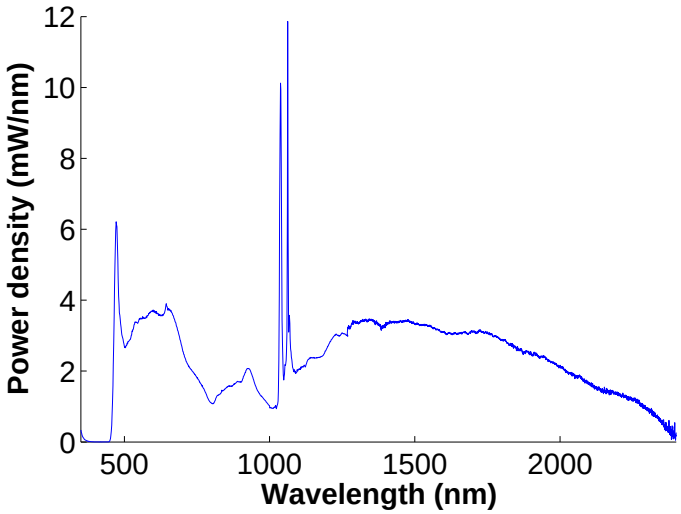


FIGURE 3.2: Power density of the supercontinuum laser.

A SPAD is a device that triggers an avalanche of current when a single photon strikes the diode surface. When the sample fluoresces, the emitted light scatters through the cuvette in all directions. We position the SPAD on the output port of the cuvette holder. Light leaving this port is filtered and focused onto the SPAD's aperture. The SPAD has a timing resolution of 40ps. The output of the SPAD is

sent to the time-correlated single photon counter (TCSPC) module attached to a computer. The TCSPC measures the cumulative number of photons that strike it within the time interval between two consecutive laser pulses as well as the time of detection within this interval. It is synchronized with the laser's pulses via a sync signal provided by the laser. Various operating parameters of the TCSPC, such as time scaling, time offsets, and sensitivity levels, are configurable through a software package on the computer. We refer the reader to the SPAD manual for operational details [5].

3.2 Preparation

We use standard DNA self-assembly procedures to form the lattices and conjugate fluorophores. Briefly, for a 4x4 grid composed of 16 tiles, we assemble each tile independently and design the sticky ends such that it will bind only to one other tile in only one position, thereby creating a 4x4 grid. Before the tiles are combined to assemble the complete grid, we conjugate fluorophores to specific base pairs on specific tiles through 5'-SH end functionalization. Chromophores and relative distances are chosen to maximize the contribution of RET to the output fluorescence.

Samples were removed from a 5° C freezer and sonicated for 90 seconds to thaw and break up aggregates that may have formed. We split the samples into two batches, each of which then diluted to a concentration of to a total volume of 50μL. These steps were taken with care to ensure that all sample batches were exposed to light uniformly during preparation. TREX uses a special optical cuvette that holds precisely 50μL of sample in a narrow trench, which is designed for total internal reflection. Light can strike the sample through a narrow window, which is narrower than the diameter of laser beam; this ensures that the entire sample is excited equally.

3.3 Laser Alignment

TREX currently supports four channels of input. In our measurements, we used just three: one initial condition channel and two channels of excitations. For each IX block, TREX must be realigned, because the reality is that laser beams are never perfectly parallel to the physical channels. Details about laser alignment are provided in Appendix C.1.

3.4 Configuration of the TCSPC

The time-correlated single photon counter module has a plethora of parameters to set. Most of the default values can be left alone. There are several modes of operation. The default mode places photon events into time bins of an adjustable width, and this is the only information recorded. FIFO mode records both the time bin that the photon event was placed into as well as the offset within that bin. FIFO mode therefore preserves more information, and it was the mode in which our measurements were taken. In addition, the TCSPC software can choose to stop recording after a certain time has elapsed or continue on recording data in a separate file. We choose the former mode, collecting data for a total of 120 seconds per sample. Details about important parameters are provided in Appendix C.2.

3.5 Measurement

Once the sample is prepared, TREX has been aligned, the TCSPC is properly configured, and all components have been connected properly, measurement can begin. Samples that are not currently in use are stored in a dark container.

We describe the IX block delays in terms of optical breadboard holes. For instance, the IX block [5, 10, 15] means that the optical posts of the 1st, 2nd, and 3rd channels were positioned at the 5th, 10th, and 15th breadboard hole, relative to a predefined starting position. Note that delay 10 on one channel is not equal to delay 10 on another channel because of different travel times through the beam splitter network and different total travel times before striking the sample. However, this does not make any difference, because we merely require is that different IX blocks are applied identically to all the RET-PUF samples, regardless of the actual delays for each IX block.

4

Results

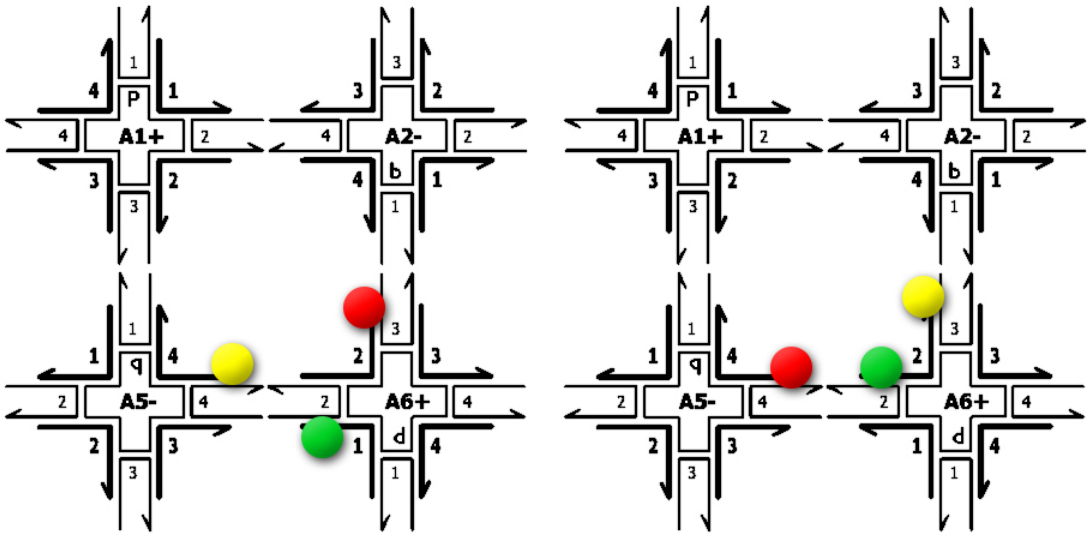
4.1 Definitions

We define the following notation for this section.

- P_A and P_B represent Alice and Bob's RET-PUFs, respectively. Alice and Bob possess identical RET-PUFs.
- r_A is shorthand for $r_A(t, \lambda)$. r_A and r_B represent responses from Alice's and Bob's RET-PUFs, respectively, to a common input.
- $r_A = r_B$ does not indicate perfect equality, but rather greatest similarity in a set of responses.
- $x_i(t)$ is the ith IX block, $1 < i < N_x$ where N_x is the number of IX blocks used (in an experiment).
- The term *input* is synonymous with *IX block*.

4.2 RET-PUF Structures

We focus our analysis on RET-PUFs 144 and 156. These RET-PUFs were chosen because they are structurally very similar to each other, yet as we will see, their fluorescence signatures are very different. The three fluorophores on each grid are Alexa Fluor 488 (green), 594 (yellow), and 647 (red). The only difference between 144 and 156 is the location of AF488, which has been shifted from shell 2 to shell 1 on the other side of arm 2. Their structures are shown in Figure 4.1.



(a) RET-PUF 144. (b) RET-PUF 156.

FIGURE 4.1: RET-PUF structures and composition.

We used 50μL of sample for each RET-PUF. Concentrations for 144 and 156 were 43.33nM and 130nM, respectively.

4.3 Setup Validation

First, we demonstrate the validity of our experimental setup by reproducing lifetime measurements of the fluorophores we will use later and showing that our measurements agree with generally accepted values. Lifetime measurements were calculated by fitting the experimental data to a decaying exponential [1].

Fluorophore	Accepted lifetime (ns)	Measured lifetime (ns)
Alexa Fluor 488	4.1	4.2
Alexa Fluor 594	3.9	3.9
Alexa Fluor 647	1.0	1.3

Table 4.1: Validation of our experimental setup [7].

4.4 RET-PUF Response

We now show the responses of multi-fluorophore systems to IX blocks consisting of multiple laser pulses at different wavelengths and delays. These responses form the

[1] Experimental data was taken by Vishwa Nellore.

building blocks of the ciphertext sent to the receiving party. Such multi-fluorophore systems must possess the following three properties:

- Two identical RET-PUFs must individually respond identically to the same input, within some noise margin.
- Two similar but different RET-PUFs must be distinguishable based on their responses.
- Two identical RET-PUFs must respond distinguishably differently to similar but different input.

To quantify the similarity between r_A and $r_B(t)$ for a given input $x(t, \lambda)$, we define a response pair distance $\mathrm{d}(r_A, r_B)$. r_A is considered identical to r_B if their distance is smaller than that between r_A and any other response. d is given by the following equation:

$$\mathrm{d}(r_A, r_B) = \sqrt{\int_t \left(\ln(r_A(t)) - \ln(r_B(t)) \right)^2} \tag{4.1}$$

For discrete data with n samples for each response, this becomes:

$$\mathrm{d}(r_A, r_B) = \sqrt{\sum_{i=1}^{n} \left(\ln(r_A(i)) - \ln(r_B(i)) \right)^2} \tag{4.2}$$

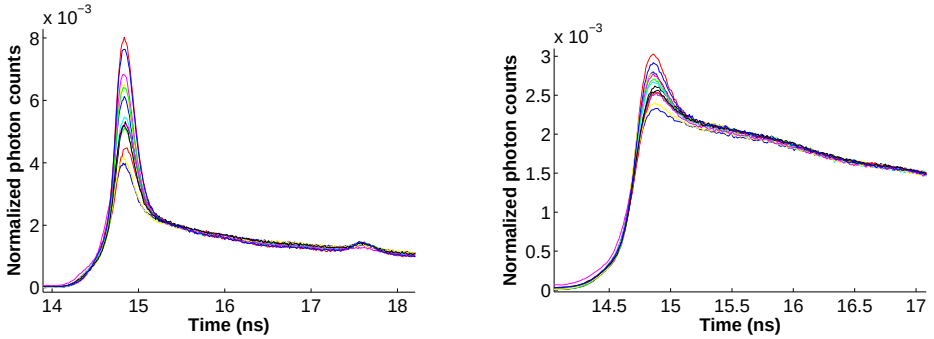
We selected 14 different IX blocks $x_1(t)...x_{14}(t)$, specified in Table 4.2 for our experiments. Figure 4.2 shows the responses for RET-PUFs 144 and 156 to these 14 inputs. To characterize how closely two copies of the same RET-PUF match, we divided each RET-PUF sample into two batches, applied the inputs to them separately, and then computed the distance d between the responses from the two batches. Figure 4.3 shows these distances. The x axis spans the 14 responses from batch 1, and the y axis spans the 14 responses from batch 2. For instance, the bar corresponding to batch 1, response 10 and batch 2, response 5 is the distance between those two responses. Lower distances denote a closer match.

Figures 4.3(b) and 4.3(d) highlight just the best matches. The ideal plot is a highlighted diagonal showing that P_A and P_B effectively represent the same RET-PUF. In PUF 144, 9 out of 14 inputs produced the same output on two batches of the same RET-PUF, and PUF 156 had 6 out of 14; these are the IX blocks that can be used to encrypt message blocks in the cryptosystem.

There are errors in the deduced matches as shown in Figure 4.3. The incorrect matches are most likely due to a flaw in the best match metric and/or a characteristic of the RET-PUF. We suspected that this was due to a flaw in the experimental procedure, but on further inspection, we found that the banding was not correlated with the order of IX blocks measured, and this phenomenon was seen with every

Number	IX Block	Number	IX Block
1	10, 10, 10	8	10, 16, 10
2	10, 10, 12	9	10, 16, 13
3	10, 10, 16	10	10, 16, 16
4	10, 13, 10	11	10, 16, 7
5	10, 13, 13	12	10, 7, 10
6	10, 13, 16	13	10, 7, 12
7	10, 13, 7	14	10, 7, 16

Table 4.2: IX blocks, in terms of breadboard holes. "Number" indicates which row and column the response matching distance is plotted in Figure 4.3.



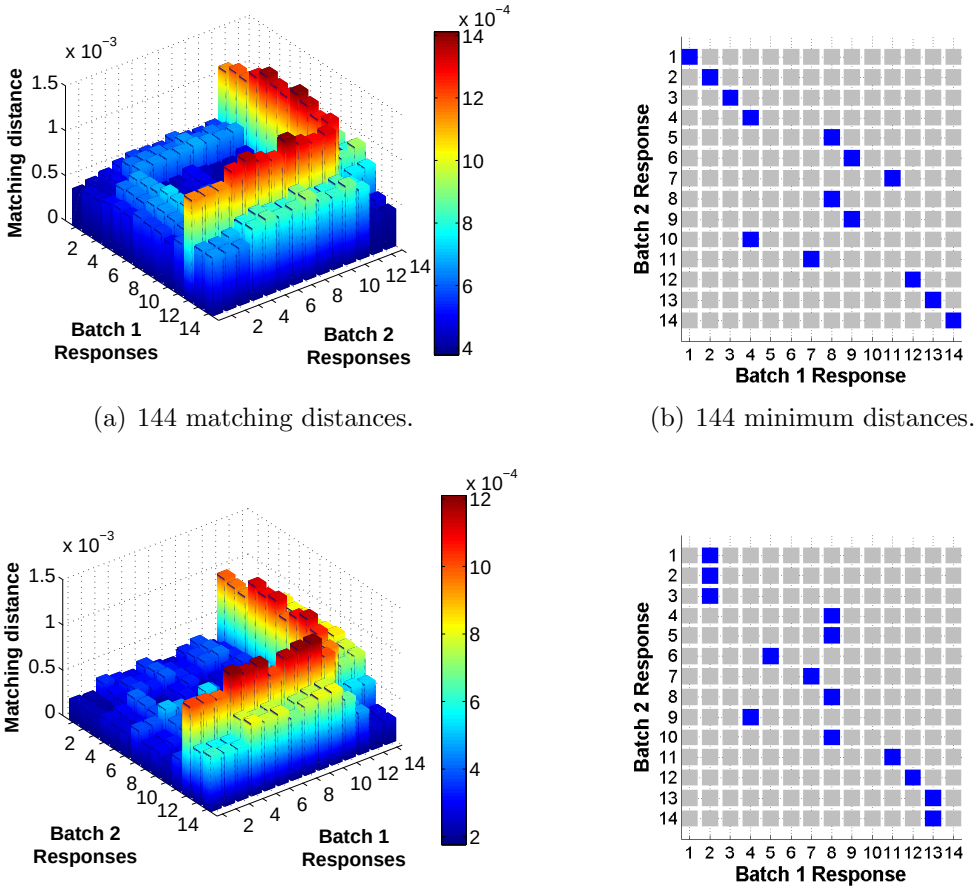
(a) RET-PUF 144 (b) RET-PUF 156

FIGURE 4.2: Fluorescence decay curves from two similar RET-PUFs.

other RET-PUF we tested. Figure 4.2 shows degeneracy, where different IX blocks appear to produce nearly identical decay curves. This seems to be the reason why some IX blocks cannot be distinguished from others. Hence, we introduce the notion of a "bad IX block" - a sequence of laser pulses which, when applied to a certain RET-PUF, cannot be reproduced reliably. it may be possible to tweak the distance metric to distinguish between such small differences. For now, the solution to this problem is simply not to use bad IX blocks in the cryptosystem.

The matching distances in Figure 4.3 were computed using the entire fluorescence decay curve. However, in the cryptosystem, we defined an observation time window from which a portion of the decay curve will be extracted and sent from Alice to Bob, representing an encrypted block of text. In order for Bob to decode the message correctly, r_A and r_B must be effectively identical during that observation window rather than over the entire decay curve. Figure 4.4 shows the lowest response pair distances for the two RET-PUFs over a 4ns window.

Intuitively, we expect that as the observation window length shrinks, the accuracy of the best match metric will degrade as there is much less data to compare, and vice versa. Interestingly, Figure 4.4 shows that using a time window can actually increase the number of correctly deduced matches between r_A and r_B; in the case of



(a) 144 matching distances.

(b) 144 minimum distances.

(c) 156 matching distances.

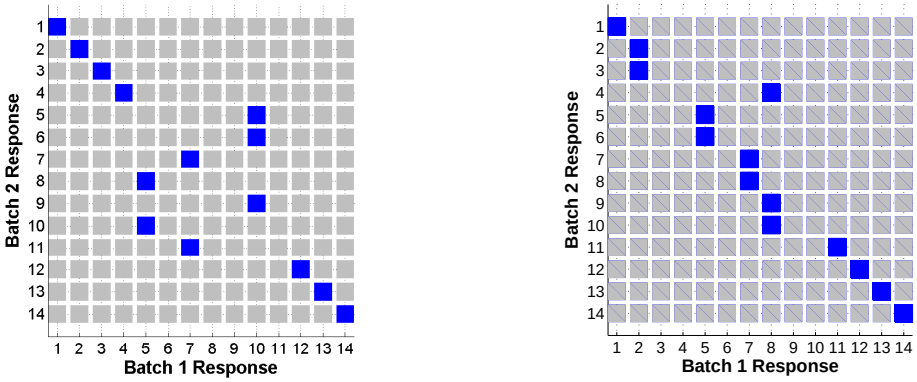
(d) 156 minimum distances

FIGURE 4.3: Response pair distances d for 14 inputs.

RET-PUF 156, that number increased from 6 to 8.

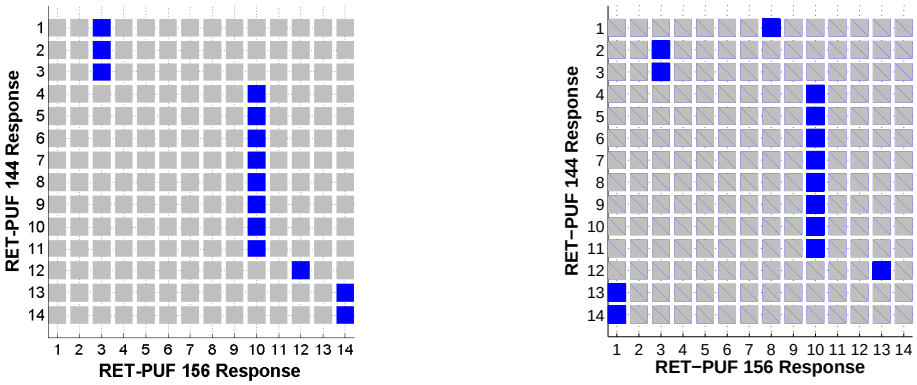
The placement of the observation window is also crucial. If the observation window is positioned along the tail end of the decay curve, many IX blocks will produce similar output, since the distinguishing fluorescent events tend to occur in the nanoseconds immediately following the input. Determining the ideal observation window length and placement is crucial to the operation of the authentication and cryptographic protocols. We found that the best observation time window for these two RET-PUFs was the window 13-17ns. This window includes the inital large peak, where the responses were highly differentiated. After around 17ns, all of the responses decayed in an identical fashion. Figure 4.2 uses this window.

Finally, we show that RET-PUF 144 and 156, despite being very structurally similar, have very distinct flourescent characteristics. We perform the same kind of analysis as before, where we compute the best match for each IX block, except we use one RET-PUF to encrypt and another to decrypt. As shown by Figure 4.5, the diagonal



(a) RET-PUF 144 (b) RET-PUF 156

FIGURE 4.4: Smallest response pair distances d for $13 < t < 17$ns.



(a) All time (b) Time windowed

FIGURE 4.5: Smallest response pair distances between two different RET-PUFs.

relationship evident on previous groups is gone, so if Alice and Bob use different RET-PUFs, Bob will not be able to deduce the input Alice applied and thus cannot decrypt the ciphertext.

Data from other RET-PUFs are presented in Appendix A.

5

Practical Use

5.1 Working with a small input space

From a practical perspective, it is much easier for Alice and Bob to perform as few measurements as needed, since that can be a time consuming process. On the other hand, fewer physical measurements means greater overheads in communication and potentially weaker security of the cipher, which we will demonstrate shortly. As technology improves, automated, miniaturized versions of our setup will greatly speed up the process. Nonetheless, the question of how many physical measurements are necessary to provide adequate security is an important one.

In Section 4, we found that in our total input space of 14 IX blocks, we found 7-9 blocks for which $R_A = R_B$. In this case, the small size of the usable input space means that directly mapping an IX block to a block of characters is not possible. In general, such a mapping is not scalable, because as the block size grows, the number of IX blocks (and thus physical measurements) that are required grows exponentially, making the cryptosystem highly impractical.

Previously, we defined an IX *sequence*, which is a permutation of a set of IX blocks. We define the "response" of an IX sequence to be the corresponding permutation of the responses of the IX blocks. As an example, suppose we have IX blocks 7, 8, and 9, which consist of the delays [1,3,5], [1,5,6], and [1,3,4] and produce responses R_7, R_8, and R_9, respectively. A sequence might be the permutation {7,9,7}, which means the underlying delays are [1,3,5,1,3,4,1,3,5] and the response the concatenation $R_T = [R_7, R_9, R_7]$[1].

[1] Note that if one were to actually use the delays comprising the sequence as the input to a RET-PUF, the measured response would not be the same as R_T.

More formally, let m be the number of usable IX blocks, n the number of valid plaintext symbols, and k the plaintext block size. m can be a small number even if n is large. In order for these m IX blocks to encode all n^k possible blocks, we need a minimum of $p = \lceil k \log_m n \rceil$ IX blocks per sequence. The total number of IX sequences is thus m^p. Essentially, we are mapping a higher radix alphabet (the plaintext alphabet) onto a lower radix alphabet (the set of usable IX blocks). We have shifted the burden of the large input space from a person (i.e. Alice) to a computer, which can compute and compare sequences very quickly.

As $m^p \geqslant n^k$, multiple distinct IX sequences may map to a single plaintext block. During encryption, one of the IX sequences will be selected at random. We will purposely pick our design parameters so that each plaintext block is mapped to by a large number of IX sequences. Such substitution ciphers, are known as *polyalphabetic* ciphers. The motivation behind this choice is to make frequency analysis more difficult, although as we will see in Chapter 6, this property itself is not enough to protect a substitution cipher from certain types of attack algorithms.

5.2 Explosion of the input space

In the previous section, we addressed the problem of an input space too large. In this section, we examine the opposite problem: how to efficiently decode a message if the input space that both parties must test is vast.

The number of total possible IX blocks m is given by the following equation. Let N_c be the number of input channels, d be the number of possible delays, and N_λ the number of available wavelengths. We can only use N_c wavelengths, and since each channel is different, there are ${}_{N_\lambda}\mathrm{P}_{N_c}$ different permutations of wavelengths on channels. Each channel can be placed at d different delays, so there are d^{N_c} possible delay permutations. Finally, we get:

$$m = ({}_{N_\lambda}\mathrm{P}_{N_c}) * d^{N_c}$$

If there are 3 input channels, 30 different delays, and 10 total wavelengths to choose from, we end up with nearly 20 million IX blocks. This is clearly impractical for either Alice or Bob; even if the fraction of valid IX blocks is small, the total number is still huge. The solution is to divide the IX block space into many smaller partitions and communicate the partition identifier in the clear. As long as we have enough partitions and enough IX sequence duplication per plaintext block, we do not compromise the security of the system. We can also divide the entire message into several regions and use a different partition of the IX block space for each region so a single partition is not overused.

To illustrate this solution, consider the (extremely) limited case when we've selected an partition of size $m = 10$, and we want to encode an input of $n = 30$ characters in

blocks of size $k = 4$. Then $p = 6$, so we have 10^9 sequences covering $30^4 = 810,000$ plaintext blocks. The number of IX sequences per plaintext block $N_s = 1.23 \rightarrow 2$, which is very low. We can remedy this by increasing p to 11, so $N_s = 137$. This raises the size of the sequence input space to 10^{11}, which is manageable for modern computers. There is no reason to limit m to 10; we can conservatively assume that $n = 30$, in which case we can achieve $N_s = 900$ with $30^6 = 720$ million different sequences. Since any alphabet used in a plaintext message can be reduced to a smaller radix alphabet (like one with $n = 30$ characters), we have therefore shown that this scalable method is feasible for practical use.

6

Cryptanalysis

6.1 Entropy

In information theory, entropy is a measure of the uncertainty of the outcome of a random variable. Claude Shannon derived the following equation to quantify the entropy H of a random variable whose k possible outputs have probabilities $p_1 ... p_k$:

$$H = -\sum_{i=1}^{k} p_i \log_2 p_i \tag{6.1}$$

There is no such thing as *the* entropy for a bit string because any measurement of entropy depends on the chosen value of k. We can choose to let $k = 2$, in which case we are simply comparing the probabilities of 0 and 1. We can let $k = 2^m$, where m is the number of bits we compare at once. We can also let $k = 2^{mn}$, in which case we are considering n-grams of m bit characters. In general, as k increases, so does the maximum entropy value. For the analysis that we will perform, we study n-grams of 8-bit characters.

We compare entropies of the ciphertext bit string generated from the RET-PUF cryptosystem to those generated by other well known cryptographic schemes. For simplicity, we chose the plaintext to be the famous Gettysburg address. We chose RET-PUF 156 as our private key, a 128-bit AES key, and a 56-bit DES key. Encryption using existing cryptosystems was accomplished using the Java cryptographic functions. We did not analyze RSA because that system is not designed to encrypt large blocks of data. Table 6.1 shows the results. For all but the smallest n-grams, the RET-PUF cryptosystem produces ciphertext with significantly higher Shannon entropy than AES or DES. Furthermore, the RET-PUF Shannon entropy steadily

increases as n grows, whereas the other cryptosystems' entropies tend to decrease and converge.

Cryptosystem	Entropy				
	1-gram	2-gram	3-gram	4-gram	5-gram
RET-PUF	7.347	11.169	13.905	12.118	14.184
AES	8.784	9.167	8.585	8.170	7.852
DES	7.822	9.167	8.585	8.170	7.852
Triple DES	7.834	9.167	8.585	8.170	7.852
Blowfish	7.840	9.167	8.585	8.170	7.852

Table 6.1: Comparison of Shannon entropies of bit strings for various cryptosystems.

6.2 Advantage Time

Here we compute the advantage time of this cryptosystem. Advantage is defined as the ratio of the amount of time an attacker would require to decipher a ciphertext via brute force to the amount of time a legitimate user would require to accomplish this task. For the RET-PUF cryptographic scheme, the private key is the RET-PUF; ownership of the RET-PUF is the only advantage that a legitimate user has over an attacker. Thus, the advantage is given by:

$$A = \frac{N_{PUFs} * t_{sim}}{t_{eval}}$$

t_{sim} is the amount of time to simulate the response for one IX block on one RET-PUF, and t_{eval} is the amount of time it takes to experimentally perform the same measurement. Notice that the advantage is independent of the number of IX blocks, because both the attacker and the legitimate user have to iterate through them all. The number of possible RET-PUFs is more difficult to accurately compute, because structural symmetries must be taken into account. However, an order of magnitude of estimate will suffice for the purposes of this argument. If m is the total number of distinct chromophores, n is the number used for this RET-PUF, and l is the number of sites on the DNA lattice, then there are m^n possible sets of chromophores - we don't enforce the restriction that all chromophores must be distinct. Each set can be arranged on the grid in ${}_l\mathrm{P}_n$ ways. Therefore,

$$N_{PUFs} \approx (m^n)({}_l\mathrm{P}_n) \tag{6.2}$$

To get an idea of how large this value is, let's assume that $t_{eval} = 1\text{s}$, a conservative estimate, and $t_{sim} = 1\text{ns}$, a generous estimate. There are around 30 different types of chromophores readily available. Let's choose 8 of them. We can tether chromophores

to any base pair on the DNA lattice, but very conservatively, we choose only 50 locations, taking into account the fact that chromophores cannot be placed arbitrarily far apart (or RET will not occur). With these numbers, $N_{PUFs} \approx 10^{25}$, and $A = 10^{16}$. If an attacker combined the computational power of 1 billion supercomputers, each of which could sweep through the entire input space for a RET-PUF in one second (which is extremely generous), it would take the attacker 10^{11} years to search all possible RET-PUF structures. This is a huge advantage, even more impressive given that our assumptions for this calculation were very conservative. The advantage calculation shows that the security of the cryptosystem rests on the difficulty of determining the RET-PUF's structure.

Of course, practical cryptographic attacks do not attempt a brute-force attack. We examine several applicable attacks on our cryptosystem in the following sections. The literature on cryptographic attacks is huge, and therefore we cannot not address them all; instead, we will address the ones that are the most applicable.

6.3 Frequency analysis

As the cryptographic protocol is fundamentally a substitution cipher, it is crucial to secure the protocol against a frequency analysis attack. In a frequency analysis attack, an attacker analyzes the frequency of characters in the ciphertext and compares the distribution of ciphertext characters with a known distribution of letters in a language. In the English language, 'e' is the most often used letter at around 12%, followed by 't' and 'a' at 9% and 8% respectively [18]. If 'm' is the most frequently used letter in the ciphertext (with a frequency of about 12%), it is highly likely that 'm' maps to 'e', and so on. In a simple substitution cipher that merely exchanges single characters with other characters from the same alphabet, the underlying frequency distribution is unchanged, making frequency analysis trivial. Other trends, like the frequency of the first and last letters, can also be used to glean information about the ciphertext.

An improvement to the simple substitution cipher is to allow multiple ciphertext characters map to a single plaintext character ('e' could be represented by either 'm', '!', 'k', etc). Such ciphers are called polyalphabetic ciphers. This alters the underlying frequency distribution, making a direct comparison with a known distribution more difficult. One category of polyalphabetic ciphers is the homophonic cipher, in which each ciphertext character appears equally often and all possible ciphertext characters are used. The resulting frequency distribution is completely flat, but such a cipher is not as secure as it might suggest, due to patterns in languages. For instance, 'q' is a rarely used letter in English, so an attacker could hypothesize that it only has one mapping (to preserve the uniform frequency distribution). Bigrams, trigrams, and larger n-grams in languages can provide additional cludes. Hill-climbing algorithms

can crack such ciphers in under a second [2].

Another well known polyalphabetic cipher is the Vigenère cipher, in which the ciphertext is generated by modular addition of the plaintext and a repeating key. For practical reasons, the key must be of a reasonably short length, but this results in periodicities in the ciphertext. An attacker can deduce the length of the key using the observed periodicities, then use this knowledge to deduce the individual bits in the key by comparing statistical properties in the ciphertext with known properties of the language of the plaintext [17]. Taken to the limit where the key is as long or longer as the message itself, the Vigenere cipher becomes the one-time pad, which has been proven to be perfectly secure [15]; however, the key must be random, as long as the message, and can't be reused, making one-time pads impractical for actual usage.

An improvement on the simple substitution cipher is to substitute blocks of characters at a time. This is known as a polygraphic substitution cipher. Polygraphic ciphers are significantly more difficult to crack than monographic ciphers for two primary reasons: the frequency distribution is inherently flatter and there are more possibilities to search through. One example of a polygraphic cipher is the Playfair cipher, invented in the 19th century, which encodes a message in bigrams. Howev, if the block size is small (in the case of the Playfair cipher it is 2), a direct frequency distribution comparison will work, because languages have distinctive bigram, trigram, and even quadrigram distributions. Moreover, there are distinctive characteristics and limitations of the Playfair cipher that make it particularly amenable to frequency analysis attacks. Brute force attacks and hill climbing techniques are slow but will succeed for small block sizes, and larger block sizes tend to make the decryption process slow.

It must be understood that any cryptographic system can be weak if its parameters are not chosen properly (short key lengths, low entropy in the key, etc.). With this in mind, we claim that the RET-PUF cryptographic protocol is immune to frequency analysis attacks because we combine a polygraphic cipher with a large block size and multiple alphabets that evolve with every message. Although an exhaustive formal proof is beyond the scope of this report, we can justify this qualitatively by explaining how the RET-PUF cryptosystem is protected against each of the attacks described above. Let Mallory be a malicious attacker.

6.3.1 Direct frequency comparison

The RET-PUF cryptosystem is a block cipher, and when blocks are sufficiently large, the frequency distribution of block usage is uniform, thus precluding a direct statistical comparison.

6.3.2 Uneven distribution of mappings

Mallory might guess that infrequently used blocks will have fewer mappings than commonly used blocks, as is the case in the homophonic substitution cipher. But in the RET-PUF cryptosystem, we assign every plaintext block an equal number of possible IX sequences, each equally likely to be used. We also divide the entire plaintext into smaller messages and change the input space partition so that each IX sequence is used on average once. This eliminates an uneven distribution of plaintext block mappings.

6.3.3 Nonuniform bitstream frequency distribution

Patterns can show up in a ciphertext's bitstream. Weak ciphers tend to have sharp spikes of commonly seen bit sequences, opening the door for a potential plaintext only attack. We analyzed the frequency distribution of bytes in the RET-PUF ciphertext by converting each value in the ciphertext to the IEEE754 double-precision floating point format, dividing the 64 bits into 8 bytes, and recording the number of occurrences of the 256 bytes. The distribution is shown in Figure 6.1.

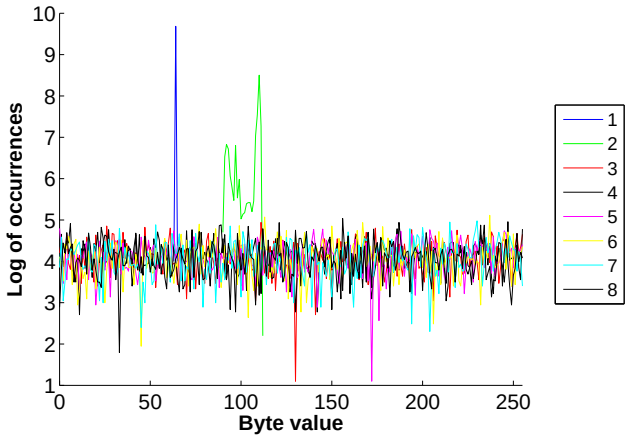


FIGURE 6.1: RET-PUF ciphertext byte frequency distribution. Legend labels indicate which lines correspond to which byte in each 64-bit floating point value. '1' indicates the first group of 8 bits, and so on.

Figure 6.1 shows several distinct peaks in the frequency distribution of the first and second bytes. The first byte peaks at 63, or '01111111' in binary. The second byte peaks from 91 to 110, or '01011011' to '01101110'. These two bytes make up the beginning 16 bits of the 64-bit value. The IEEE754 double precision format is composed of a sign bit, 11 bits of exponent, and 52 bits of mantissa. From this, we can deduce that the peaks in the frequency distribution are a result of the fact that

all the values in the ciphertext fall within the biased exponent range 2037 to 2038. Removing the bias of 1023, we get the final range 1014 to 1015.

This analysis shows that Mallory would be able to deduce the range of values that the RET-PUF ciphertext falls within. However, this is information that Mallory could have deduced *as long as she was aware that the RET-PUF cryptosystem was being used*, which is an assumption we must make. Therefore, the presence of these peaks in the ciphertext bitstream reveals little to no information that Mallory would not be able to guess otherwise.

6.3.4 Hill climbing algorithm

This algorithm relies on the fact that for substitution algorithms, the closer a putative key approaches the true key, the more closely the putative plaintext will resemble the plaintext. Since the RET-PUF is the true key, which is not applicable for frequency analysis, the secondary key that such algorithms attempt to deduce is the mapping between plaintext and ciphertext blocks. It is highly efficient if a small change induces an isolated change in the putative plaintext; that is, if 'm' is substituted by 't' and the attacker correctly changes this guess for 'm' from 'k' to 't', the frequency distribution of 'm' will more closely match the predicted fraction without negatively impacting other predictions. However, in a polygraphic system, abcd might be mapped to 't3df', and if the attacker changes the guess from 't3ee' to 't3de', the new mapping would be entirely different. Isolated changes in the ciphertext result in larger changes to the entire putative plaintext. Furthemore, use of the nonce precludes this attack as it completely eliminates any such depedence. Hill climbing algorithms are herefore highly inefficient against the RET-PUF cryptosystem.

6.4 General Attacks

6.4.1 Gradual accumulation of information

Mallory can eavesdrop on the communication session and store observed enciphered messages to build a large database of ciphertext from which she can extract trends and correlations to break the system. However, a RET-PUF evolves with every use, so each message will consist of a slightly different alphabet. Furthermore, the use of a nonce for each message block obfuscates the true message, so whatever information she manages to extract will be useless.

6.4.2 Replay attack

A replay attack occurs when Mallory stores an encrypted transmission of information from Alice and repeatedly sends it to Bob multiple times. This cryptosystem is immune to such an attack because we use a nonce to make each encrypted message valid only once. In addition, the RET-PUF evolves over time, so eventually Bob will be unable to successfully decrypt the received message, at which point he will know that there has been tampering with his communications to Alice.

6.4.3 Known plaintext attack

A known plaintext attack is one in which the attacker uses the system to generate a ciphertext from a known plaintext. The cipher is broken if the attack is able to glean a relationship between the ciphertext and the plaintext, for this is information that the attacker can potentially use to determine the secret key. The RET-PUF cryptosystem is immune from this type of attack. In order to mount this attack, Mallory must physically steal the RET-PUF without detection. Although this is not physically impossible, Mallory must either use the RET-PUF before the owner realizes it is missing or remove a fraction of it for later use. If Mallory uses the RET-PUF and then returns it, subsequent communications between Alice and Bob will fail because one RET-PUF has aged while the other has not, and this discrepancy will show. If Mallory chooses to steal a fraction of the sample for later use, she must take a non-insignificant amount for the stolen sample to produce reliable results, and this is easily detectable. If, in the future, the attacker could steal a single DNA grid and produce the same results, we could simply modify the cryptoprotocol to use a single grid instead of a population. Due to RET-PUF aging, a large database of challenge-response pairs is ineffective, and so this form of attack is also negated.

6.5 Physical compromise

The RET-PUF is immune against physical attack due to the nature of the RET-PUF itself and the protocol in which it is used. We discuss a few possible attack vectors in this section.

6.5.1 Dilution attack

The response of the RET-PUF is highly dependent on parameters such as concentration, pH, ambient temperature, etc. If Mallory were to steal an appreciable amount of sample, the difference in volume would easily be detectable. If Eve were to dilute the remaining sample back to the original volume, she would alter the concentration

of the RET-PUF, also easily detectable. If this is taken to the limit, Eve would steal a single RET-PUF, undetectable through ordinary measurements. However, single grid measurements produce much more noise and are much more difficult to consistently duplicate accurately. Nonetheless, if single grid measurements were feasible, the RET-PUF cryptosystem can thwart this by using only single grids, preventing Mallory from stealing and using the RET-PUF without detection.

6.5.2 Super-resolution microscopy

Super-resolution microscopy is another attack vector. Ordinary light microscopy is limited to resolving details of around $\approx$ 250nm [1]. Super-resolution microscopy allows much smaller details to be discerned, either through capturing information in evanescent waves or through clever experimental techniques that enables reconstruction of nanoscale details below the 250nm limit.

Near-field scanning optical microscopy is a technique that places a probe very close to the surface of the object to capture evanescent waves emitted from the surface. This distance must be much smaller than the wavelength of light behing emitted. Research in this field has produced techniques capable of resolving lateral details to around 20nm [10]. However, these techniques are still an order of magnitude away from resolving details on the order of 1-5nm, which is the level of detail needed to break a RET-PUF.

Clever experimental methods have been devised to capture nanostructure details at resolutions of around 10nm, like PALM [24]. Stimulated emission depletion (STED) microscopy is a super-resolution technique developed in 1994 [4] that works by recording the fluorescence of a population of dyes over time as individual dyes are selectively photobleached. The change in fluorescence intensity enables one to determine the center of the fluorescence region, corresponding to the location of the dye. STED can resolve details with 35nm accuracy; clearly, such a technique would not be able to break the RET-PUF's security.

Fluorescence imaging with one nanometer accuracy (FIONA) fits the fluorescence of a single dye with a two-dimensional Gaussian to resolve the position of a single dye molecule with 1nm precision [23]. However, for a population of dyes, it can only discern two dyes separated by at least 6.5nm. This is roughly the Förster radius, and for cryptographic purposes, fluorophores would be placed within this radius. FIONA is thus unable to resolve the arrangement of fluorophores in this case - the best it can do is determine the center of the population of dyes.

These are just a few of the many super-resolution techniques currently available. Many of them use the same principle: selectively activate or bleach fluorophores and resolve their positions over time. Even though no super-resolution technique using this principle can currently resolve inter-molecular separations of around 1nm, we

can combat such attacks by using two or more of the same dyes on the RET-PUFs so that selectively activating or bleaching individual fluorophores is impossible.

6.6 Self-revocation

Due to photobleaching effects, after a certain number of uses, photobleaching effectively reduces the RET-PUF to a smaller network of chromophores, which compromises the security of the system. Thus, after this point, both parties must discard this RET-PUF and switch to a new one. The nature of the RET-PUF forces its own revocation after a specified number of uses. This precludes the possibility of an attack that gradually builds up a database of challenge-response pairs for a given RET-PUF and uses that database to decipher transmitted ciphertext. The length of time after which revocation occurs can be modulated with anti-photobleaching agents that prolong the life of a fluorophore. This is an important property of cryptographic keys that is inherent to the RET-PUF but must be enforced externally upon other cryptosystems.

7

Conclusion

We have demonstrated a novel physical unclonable function based on resonance energy transfer, called a RET-PUF. The RET-PUF is constructed by embedding fluorophores on a self-assembled DNA lattice, in which resonance energy transfer (RET) is the heart of the encryption mechanism. The RET-PUF possesses a trapdoor property that enables near-perfect duplication by a party that knows the precise location of the fluorophores on the grid. Therefore, it can be distributed to multiple parties, enabling secure communication without needing to physically transport it.

We showed that two identical RET-PUFs respond very similarly to the same input and distinguishably differently to different inputs. However, two similar but different RET-PUFs respond almost entirely different regardless of the input.

The RET-PUT design space is huge. Conservative estimates place the number of distinct RET-PUFs on the order of 10^{25}, and an attacker having all the computational power in the world would require at least 10^{11} years to break the cryptosystem via brute-force. The critical features of the RET-PUF are on the order of 1nm or less, making it impervious to any known super-resolution microscopy technique. It is infeasible for an adversary to physically tamper with or steal the RET-PUF without being detected.

Furthermore, the RET-PUF suffers from photobleaching effects, which causes its response to evolve over time and sets a cap on its useful lifetime. This provides an additional layer of protection from a replay attack, because two responses observed from the same stimulus applied at different points in time will not be the same. By the same logic, the RET-PUF will revocate itself after a certain number of uses, thereby solving a problem inherent to many existing cryptosystems with the RET-PUF's intrinsic properties.

We designed a simple encryption and decryption algorithm, based on a substitution cipher that uses the RET-PUF as an intermediary layer between the plaintext and the ciphertext. This cryptosystem is impervious to frequency analysis attacks as well as a variety of other general attacks. It does not rely on number theory for encoding data, and thus a large body of work describing number theoretic attacks on cryptosystems like RSA, AES, DES, etc. do not apply.

The RET-PUF is a very promising cryptographic device that holds many implications in the security domain. We envision that the RET-PUF could be embedded in products like smart cards and USB keys as well as used in dedicated hardware for secure communication. However, for this vision to come to fruition, there are several major practicalities that stand in the way, and these comprise the avenues of future work we would like to pursue. First, we want to explore data stream compression. The RET-PUF cryptosystem results in a large amount of overhead per bit; however, the ciphertext bits are amenable to compression and in doing so we could significantly reduce this overhead. Second, we want to shrink the size of TREX, automate the alignment process, and maximize the photon count rate measured by the SPAD. This can greatly expedite the measurement phase of the cryptosystem. Third, we need to continue exploring the input space of the cryptosystem, both in terms of possible RET-PUF structures and IX blocks. This will enable us to understand the probability of cryptographic collisions as well as characterize RET-PUF aging. Finally, we would like to explore different ways of safely storing the RET-PUF. In our current experimental setup, the solvent is prone to evaporation due to constant laser exposure, and fluorescent photons may scatter away from the SPAD. We envision a setup in which RET-PUFs are embedded in a transparent medium that can be easily shielded from ambient light and inserted into TREX for measurement.

Appendix A

Additional Data

A.1 Other RET-PUFs

A.1.1 Discussion

Here we present data from other RET-PUFs. For IX block configurations, I.C. stands for initial condition. All channels are given in terms of the number of optical breadboard holes away from the front of the deflecting mirror assembly. The "number" column indicates which row and column the response is graphed on the matching distance plots.

RET-PUFs 114 and 116 were the first RET-PUFs we tested, so we used fewer IX blocks than the others. Figures A.2(b) and A.4(b) show that the distances between IX blocks appear to be grouped around three values, leaking information about the IX blocks. We believe this is because the initial condition channel is too time-separated from the other channels. Accordingly, we choose initial conditions that were closer in time to the other channels for the other RET-PUFs - 144, 156, and 149, and the matching distance figures show that this regular discretization vanishes. For all RET-PUFs, the matching distance improves after using an appropriate time window.

A.1.2 Structures, IX Blocks, and Response Characteristics

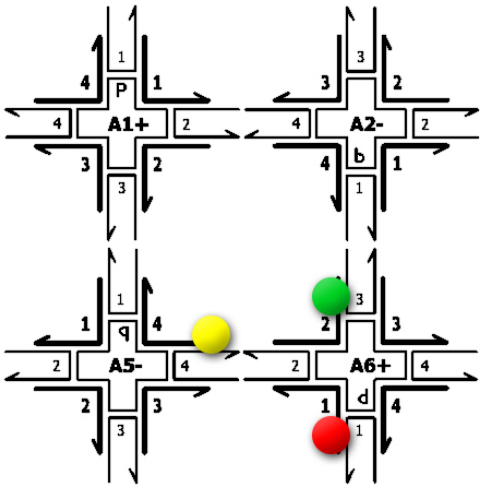
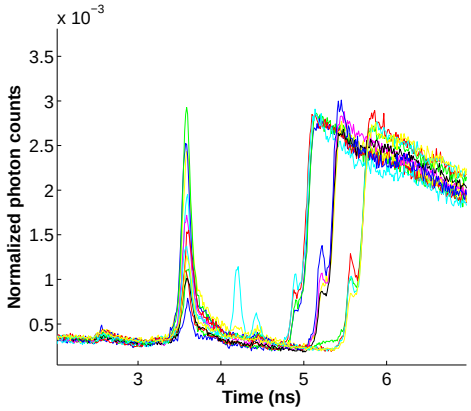


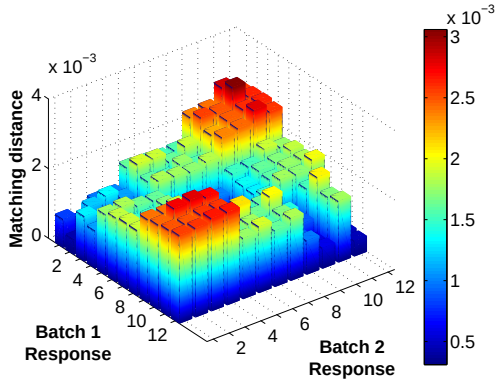
FIGURE A.1: RET-PUF 114 structure.

No.	IX block
1	0, 10, 10
2	0, 10, 12
3	0, 10, 14
4	0, 10, 16
5	0, 12, 10
6	0, 12, 12
7	0, 12, 14
8	0, 12, 16
9	0, 14, 10
10	0, 14, 12
11	0, 14, 14
12	0, 14, 16

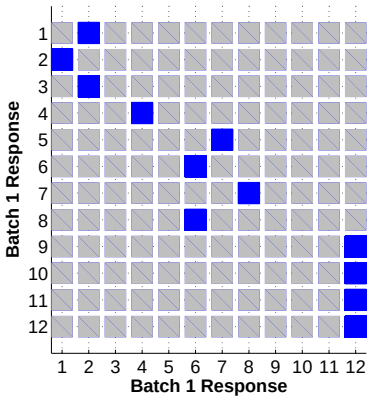
Table A.1: IX blocks.



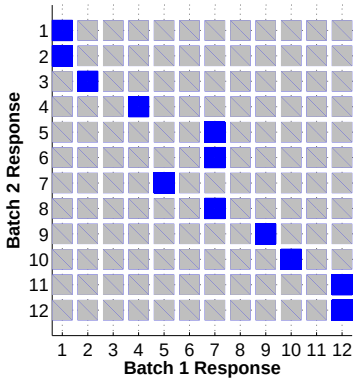
(a) Fluorescence decay curves



(b) Response matching distances

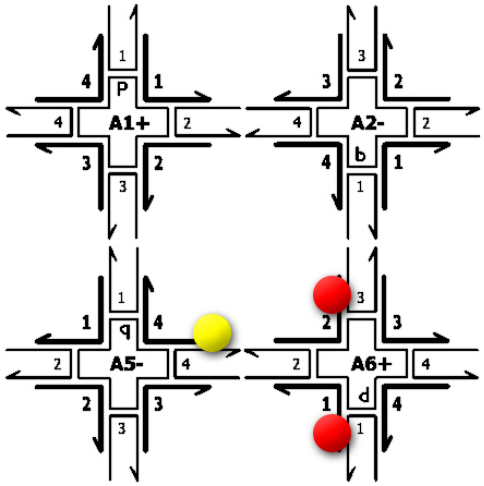


(c) Min. matching distances over all time



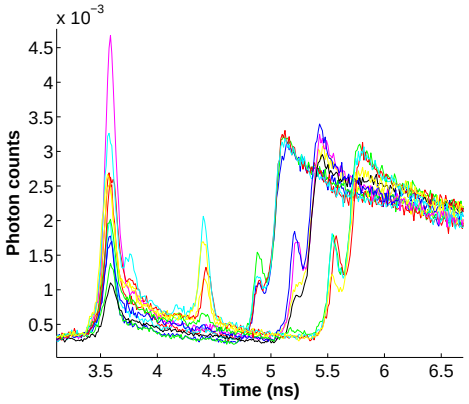
(d) Min. distances over time window 3-6ns

FIGURE A.2: RET-PUF 114 response characteristics.

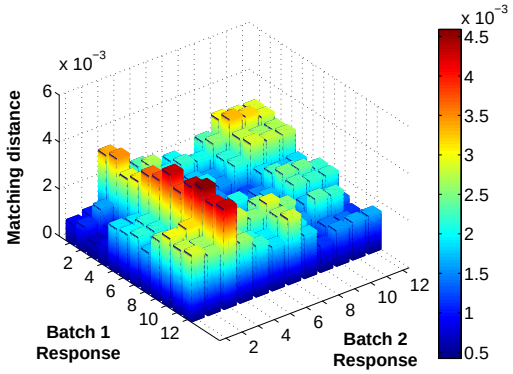


No.	IX block
1	0, 10, 10
2	0, 10, 12
3	0, 10, 14
4	0, 10, 16
5	0, 12, 10
6	0, 12, 12
7	0, 12, 14
8	0, 12, 16
9	0, 14, 10
10	0, 14, 12
11	0, 14, 14
12	0, 14, 16

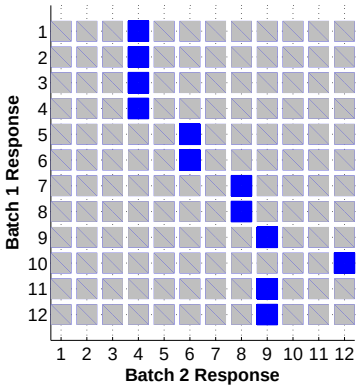
FIGURE A.3: RET-PUF 116 structure. Table A.2: IX blocks.



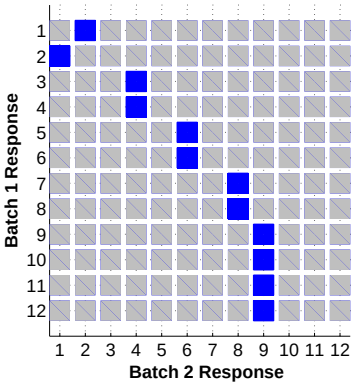
(a) Fluorescence decay curves.



(b) Response matching distances.



(c) Min. matching distances over all time.



(d) Min. matching distances over time window 3-6ns.

FIGURE A.4: RET-PUF 116 response characteristics.

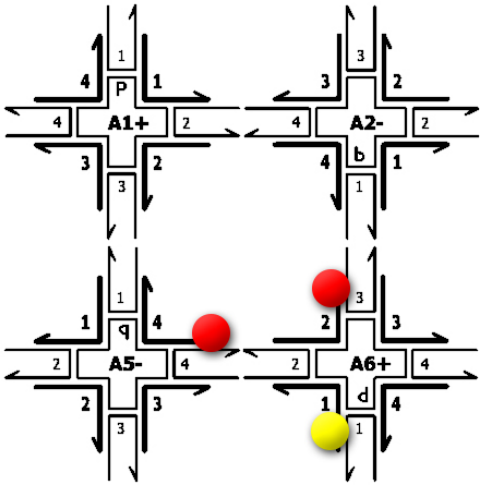
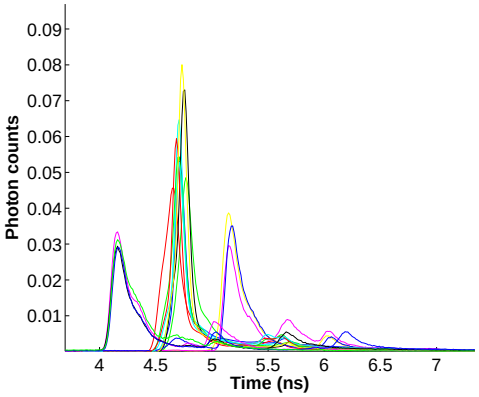


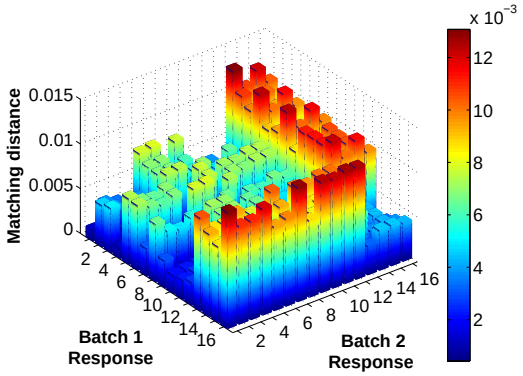
FIGURE A.5: RET-PUF 149 structure.

No.	IX block	No.	IX block
1	8, 10, 10	9	8, 16, 10
2	8, 10, 13	10	8, 16, 13
3	8, 10, 16	11	8, 16, 16
4	8, 10, 7	12	8, 16, 7
5	8, 13, 10	13	8, 7, 10
6	8, 13, 13	14	8, 7, 13
7	8, 13, 16	15	8, 7, 16
8	8, 13, 7	16	8, 7, 7

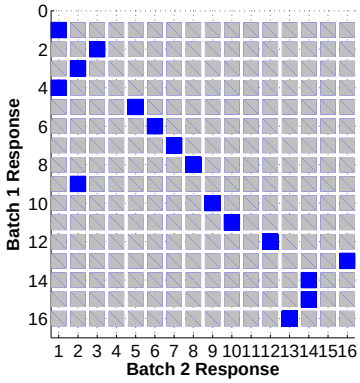
Table A.3: IX blocks.



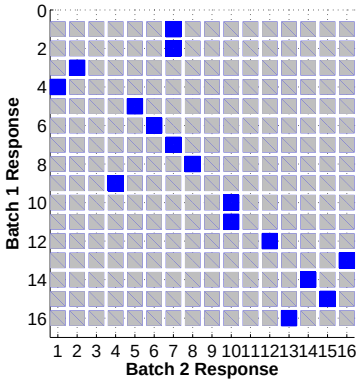
(a) Fluorescence decay curves.



(b) Response matching distances.



(c) Min. matching distances over all time.



(d) Min. matching distances over time window 4-7ns.

FIGURE A.6: RET-PUF 149 response characteristics.

Appendix B

Analysis Methods

This chapter describes how to use our code to reproduce our results. Source code is located at `www.github.com/xyzsam/RET_PUF`.

B.1 Java Crypto Functions

We used the Java Crypto library to encrypt the Gettysburg address (used in our entropy analysis) with AES and DES. Below, we describe the code that transformed the plaintext into each of the ciphertexts.

B.1.1 Plaintext manipulations

Five versions of the Gettysburg address currently exist, and there exists controversy over which version Abraham Lincoln actually delivered on that day. Of the five, the Bliss Copy is now regarded as the standard version, because unlike the others, Lincoln titled, signed, and dated this copy. The Bliss Copy, taken from Wikipedia, is provided below [8].

> Four score and seven years ago our fathers brought forth on this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal.
>
> Now we are engaged in a great civil war, testing whether that nation, or any nation, so conceived and so dedicated, can long endure. We are met on a great battle-field of that war. We have come to dedicate a portion of that field, as a final resting place for those who here gave their lives

Cipher	Key Size (bits)	Mode	Padding
AES	128	ECB	NoPadding
DES	56	ECB	NoPadding
Triple-DES	168	ECB	NoPadding
Blowfish	128	ECB	NoPadding

Table B.1: Java crypto library function parameters.

> that that nation might live. It is altogether fitting and proper that we should do this.
>
> But, in a larger sense, we can not dedicate, we can not consecrate, we can not hallow this ground. The brave men, living and dead, who struggled here, have consecrated it, far above our poor power to add or detract. The world will little note, nor long remember what we say here, but it can never forget what they did here. It is for us the living, rather, to be dedicated here to the unfinished work which they who fought here have thus far so nobly advanced. It is rather for us to be here dedicated to the great task remaining before usthat from these honored dead we take increased devotion to that cause for which they gave the last full measure of devotionthat we here highly resolve that these dead shall not have died in vainthat this nation, under God, shall have a new birth of freedomand that government of the people, by the people, for the people, shall not perish from the earth.

To simplify the encryption process, we stripped all punctuation, whitespace, and line delimiters from the text and converted all characters to uppercase. Since all of the ciphers are block ciphers, we pad the last block with a sequence of the character 'A'. This is not meant to provide any additional cryptographic security. As is described in the next section, this was the simplest way to pad the plaintext for the RET-PUF cryptosystem, and we want to compare the different ciphers on as equal terms as we can.

B.1.2 Cryptosystem Parameters

We evaluated the RET-PUF cryptosystem against AES, DES, Triple DES, and Blowfish. We used the Java cryptography library to perform the encryption, and thus the system parameters we chose were constrained by what the Java implementation provided through the SunJCE provider. In all cases, we chose the strongest parameters that the SunJCE provider allowed. Table B.1 shows the set of system parameters. The code is shown below.

```
1 import java.io.FileInputStream;
2 import java.io.FileOutputStream;
```

```
3
4  import java.security.Key;
5  import java.security.KeyPair;
6  import java.security.KeyPairGenerator;
7  import java.security.Security;
8
9  import javax.crypto.Cipher;
10 import javax.crypto.CipherInputStream;
11 import javax.crypto.CipherOutputStream;
12 import javax.crypto.KeyGenerator;
13 import javax.crypto.SecretKey;
14
15 public class Cryptosample {
16
17   public static void main(String[] args) throws Exception
        {
18
19     String algorithm = "AES";
20     Cipher cipher = Cipher.getInstance(algorithm);
21     KeyGenerator keygen = KeyGenerator.getInstance(
         algorithm);
22     keygen.init(128);
23     SecretKey secretKey = keygen.generateKey();
24
25     // RSA
26     // KeyPairGenerator keypairgen = KeyPairGenerator.
         getInstance(algorithm);
27     // keypairgen.initialize(1024);
28     // KeyPair pair = keypairgen.generateKeyPair();
29     // Key publicKey = pair.getPublic();
30     // Key privateKey = pair.getPrivate();
31
32     cipher.init(Cipher.ENCRYPT_MODE, secretKey);
33
34     String cleartextFile = "gettysburg2.txt";
35     String ciphertextFile = "ciphertext"+algorithm+".txt";
36
37     FileInputStream fis = new FileInputStream(
         cleartextFile);
38     FileOutputStream fos = new FileOutputStream(
         ciphertextFile);
39     CipherOutputStream cos = new CipherOutputStream(fos,
         cipher);
```

```
40
41      byte[] block = new byte[8];
42      int i;
43      while ((i = fis.read(block)) != -1) {
44        cos.write(block, 0, i);
45      }
46      cos.close();
47
48      String decryptedTextFile = "decrypted"+algorithm+".txt
           ";
49      cipher.init(Cipher.DECRYPT_MODE, secretKey);
50
51      fis = new FileInputStream(ciphertextFile);
52      CipherInputStream cis = new CipherInputStream(fis,
           cipher);
53      fos = new FileOutputStream(decryptedTextFile);
54
55      while ((i = cis.read(block)) != -1) {
56        fos.write(block, 0, i);
57      }
58      fos.close();
59    }
60  }
```

B.2 MATLAB Analysis

We wrote a substantial amount of MATLAB code to process and analyze RET-PUF data. The code is available on a GitHub repository located at: `http://www.github.com/xyzsam/RET_PUF`. The vast majority of the code is thoroughly documented. Here, we will give a brief overview of the functionality of the important scripts.

The codebase is divided into three libraries: `analysis`,`crypto`, and `util`. `analysis` contains functions to compute RET-PUF correlation coefficients and entropy on ciphertexts, as well as perform frequency analysis on ciphertexts. `crypto` contains functions to perform encryption and decryption. `util` contains a variety of utility functions, including visualization of analysis data and conversion of raw data into formats amenable for MATLAB.

In this section, we will demonstrate basic usage of the important functions. Since all functions are thoroughly documented, we will not go into excessive detail for every one.

B.2.1 Data Preparation

Fluorescence data is obtained from TREX in a `.asc` file format. The function `util.asc2mat` converts a `.asc` file to a MATLAB binary file. It assumes that the user has used the following naming convention for the `.asc` file, from which it parses out information relevant to the particular file:

sa[N]_ic[IC]_i[INPUT]_em[WAVELENGTH].asc

where N is a RET-PUF identifying number, IC is the initial condition delays, INPUT is the input sequence, and WAVELENGTH is the observation wavelength. Delays are given in terms of optical breadboard holes. Hence, the file `sa114_ic0_i14,14_em620.asc` indicates that RET-PUF 114 is used with the initial condition channel at hole 0, input channels both at holes 14, and an output filter of 620nm.

Other parameters to `util.asc2mat` are the total time interval between laser pulses and the width of each time bin. Both of these parameters are system parameters of the TCSPC software package.

B.2.2 IX Sequence to Plaintext Mapping

In order to encrypt and decrypt data, we need to generate a table that maps IX sequences to blocks of plaintext characters. The function `crypto.generate_mappings` performs this task automatically. It maps a sequence of k out of n possible characters to a sequence of p out of m possible IX sequences, where $p \geqslant \lceil k \log_m n \rceil$. As an example (see the code below), suppose our plaintext contains only n=26 distinct characters, and we want to encrypt this in blocks of k=4. If our experimental session yields $m = 9$ good IX sequences, then $p \geqslant 6$, and thus `crypto.generate_mappings` returns an array of dimensions (9^6) x $(4 + 6) = 53441$ x 10. The first 4 columns represent the plaintext character sequence, and the remainder represents the mapped IX sequence. There are more IX sequences than there are plaintext blocks, and so multiple IX sequences will map to the same plaintext block. Mappings are randomly permuted every time this function is run so there is no predictable pattern.

```
mappingdb = crypto.generate_mappings(26, 4, 9);
save mappingdb.mat mappingdb;\footnote{It is a good idea
   to save this array for later usage so that it does not
   have to be regenerated every time.}
```

B.2.3 Encryption and Decryption

Encryption and decryption are handled by the two functions `crypto.encrypt` and `crypto.decrypt`. Their usage is demonstrated below.

First, we initialize some required parameters.

```
data_dir = './measurements/encrypt';
puf_type = 'sa144';
output_wavelength = 670;
initial_condition = 0;
time_window_center = 15e-9;
time_window_width = 2e-9;
```

Then, we load our mapping table (if necessary).

```
load('mappingdb.mat');
```

We load our plaintext file or string.

```
plaintext = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';
% --OR--
plaintext = fileread('./plaintext.txt');
```

Now, we call `crypto.encrypt`. The ciphertext is returned as an array of floating point values.

```
ciphertext = crypto.encrypt(plaintext, mappingdb,
   time_window_center, initial_condition,
   output_wavelength, data_dir, puf_type,
   time_window_width);
```

To decrypt, we need to first provide a different folder that contains the fluorescence histograms we want to use for decryption (you could use the same but that would defeat the purpose as Alice and Bob would clearly not produce the exact same fluorescence curves). Then we can call `crypto.decrypt`. The function signatures are almost exactly the same, differing only in that one takes a plaintext string and the other takes an array of floating-point values.

```
data_dir = './measurements/decrypt';
decrypted_plaintext = crypto.decrypt(ciphertext, mappingdb
   , time_window_center, initial_condition,
   output_wavelength, data_dir, puf_type,
   time_window_width);
```

The decryption process can take some time, depending on the size of the mapping table and the computational power of the computer being used. MATLAB is not an optimal programming language for fast numerical computation, and this implementation does not take advantage of the many data parallelisms present. This process could be enormously accelerated by using GPUs to perform the computation, but this is not the focus of our work.

You have now encrypted a plaintext and decrypted the ciphertext. Now, we can discuss ways of analyzing the ciphertext.

B.2.4 Matching Distance Analysis

All of the functions used to analyze fluorescence and ciphertext data are contained under the `analysis` and `util` namespaces. We will discuss a few in this section.

`analysis.batch_intrapuf_analyze_dir` compares two sets of fluorescence measurements over a set of IX blocks in a directory. The function takes the location of the main data directory and an operation mode as arguments. The main data directory must be composed of two subfolders named '1' and '2'. Each subfolder must contain an equal number of `.mat` files, which are fluorescence measurement files produced by the function `util.asc2mat`. The mode argument indicates the comparison function and the type of data to be compared. Several comparison functions are available, but the one usually used is `ssd`, short for "sum of squares of differences". Similarly, several types of data are available, but the most commonly used type is `loghist`, which means to take the log of the raw fluorescence histogram. The mode argument is formed by joining these two strings with an underscore. The function returns an n x n matrix, where n is the number of IX blocks (and therefore files in each subfolder). We used this function to produce the matching distance figures in the Results section.

Here is a code example.

```
>> data_dir = './measurements/session1';
>> mode = 'ssd_loghist';
>> result = analysis.batch_intrapuf_analyze_dir(data_dir,
   mode);
Mapping of IX block number to the actual IX block.
1: 10,10
2: 10,12
3: 10,14
4: 10,16
5: 12,10
6: 12,12
7: 12,14
8: 12,16
9: 14,10
10: 14,12
11: 14,14
12: 14,16
>> result(1,1)
```

```
ans =

  8.2258e-004

>> result(1,2)

ans =

  6.1088e-004
```

The output produced by the function indicates which IX block each row and column of the `result` matrix refer to. `result(1,1)`=$8.2258 * 10^{-}4$, is the correlation coefficient between two fluorescence decay curves produced by the same RET-PUF with the 1st IX block, [10, 10]. `result(1,2)` is smaller than `results(1,1)`, which makes intuitive sense as now we are comparing fluorescence decay curves from two different IX blocks, [10, 10] and [10, 12].

We can also analyze this kind of fluorescence data over a time window. The function `analysis.batch_windowing_analyze` performs this task. Its function signature and behavior is almost identical to `analysis.batch_intrapuf_analyze_dir`, except it takes two extra parameters: the start and end of the time window in nanoseconds.

```
data_dir = './measurements/session1';
mode = 'ssd_loghist';
start_time = 4; % in nanoseconds
end_time = 7;
result = analysis.batch_windowing_analyze(data_dir, mode,
   start_time, end_time);
```

We can visualize this data in two ways: a 3D histogram chart showing the correlation coefficients between all fluorescence histograms in these two sets of measurements, or a flat 2D chart that shows the best matches between the histograms (i.e. the lowest correlation coefficient in a row). `util.bar3_color` produces the former, and `util.bar3_relcolor` produces the latter. Both take one parameter: the `result` matrix returned by one of the analysis scripts described above. See Figure 4.3 for examples of both types of graphs.

B.2.5 Other Analysis

Computing the Shannon entropy of RET-PUF ciphertext is handled by the function `analysis.entropyn`. The function groups the 64 bits in each double precision floating-point number into 8-bit chunks, forms n-grams from those chunks, and tallies

the number of times each such n-gram appears in the ciphertext bit stream. Shannon entropy of the bitstream is then calculated by the usual equation, where p_i is the probability of the ith n-gram. The two parameters to the function are the ciphertext array and the value of n. Since it is straightforward to use, we will not show a code example.

`analysis.freq_analysis` plots the frequency distribution of 8-bit groups in the ciphertext bit stream, taking the ciphertext array as its one parameter. This is also straightforward to use so we will not show a code example. This function was used to generate Figure 6.1.

Appendix C

Experimental Procedures: More Details

C.1 Laser Alignment

Setting TREX for a certain IX block is done through the following steps:

1. Shift the optical posts to the desired delay. Optical posts provide an accuracy of about 0.06ns per cm of shift, because the light makes a roundtrip down the channel and back. Keep in mind that each channel has a different baseline delay because some channels require the laser beam to be deflected multiple times before exiting the splitter network.
2. Use the longitudinal translation stage to finely adjust the delay of the channel. As stated before, one full rotation of the adjustment dial corresponds to 6.6ps of delay.
3. Verify the delays by collecting an empty measurement. In the absence of a sample, the SPAD will simply detect the excitation pulses.

During this process, TREX must be aligned so that all the beams land on the cuvette observation window. Alignment of TREX consists of the following steps:

1. Adjustment of the beam splitters and beam deflectors angles, so that the laser beams emerge out of the splitting network close to parallel to the tracks on which the optical posts are mounted.
2. Adjustment of the lateral position of the Dove prisms on the optical post, using the lateral translational stage, to ensure that the laser beam enters on one side and emerges cleanly through the other.

3. Adjustment of the right angle mirrors to direct the laser beam from the Dove prisms into the narrow window of the cuvette.

C.2 TCSPC Parameters

The details of most of the parameters are left to the manual of the TCSPC being used. We describe a few key parameters that are conceptually simpler to grasp and more immediately useful

- Gain: this is a scaling factor on the measured time delay of a photon event. Increasing the gain will elongate the fluorescence decay curve, while decreasing it will have the opposite effect.
- Range: this is the time range over which the TCSPC will record data. If this time range is very short (less than 10 ns), data loss is possible, as most fluorophores have excited-state lifetimes of several nanoseconds.
- Offset: this creates a time delay in the recorded fluorescence decay curve. Since TREX creates a delay between the start of a laser's pulses and those pulses striking the sample, this offset can be useful in making the fluorescence appear to start at time 0, where time 0 is the time of the laser sync pulse.
- Collection time: as the name suggests, this determines the length of time of each collection. The longer the collection time, the more photon counts we will obtain, and the more accurate the data will be. However, if the collection time is too long, photobleaching effects can hamper subsequent measurements and even later counts in the same collection.

Bibliography

[1] Abbes, E. (1873), "Beiträge zur Theorie des Mikroskops und der mikroskopischen Wahrnehmung," *Archiv für mikroskopische Anatomie*, 9, 413–418.

[2] Dhavare, A. (2011), "Efficient Attacks on Homophonic Substitution Ciphers," Master's thesis, San Jose State University.

[3] Eisenfeld, J. and Ford, C. (1979), "A Systems-Theory Approach to the Analysis of Multiexponential Fluorescence Decay," *Biophysical Journal*, 26, 73–83.

[4] Hell, S. W. and Wichmann, J. (1994), "Breaking the diffraction resolution limit by stimulated emission: stimulated-emission-depletion fluorescence microscopy," *Optics Letters*, 19, 780–782.

[5] IDQ (2012), *id100 Series Single-Photon Detectors*, IDQ.

[6] Inc., T. (2012), *1/2" Dovetail Translation Stage*, Thorlabs Inc.

[7] ISS (2012), "Lifetime Data of Selected Fluorophores," Web.

[8] Lincoln, A. (1863), "The Gettysburg Address - Bliss Copy," Web, `http://www.abrahamlincolnonline.org/lincoln/speeches/gettysburg.htm`.

[9] McAndrew, A. (2011), *Introduction to Cryptography with Open-Source Software*, CRC Press.

[10] Oshikane, Y., Kataoka, T., Okuda, M., Hara, S., Inoue, H., and Nakano, M. (2007), "Observation of nanostructure by scanning near-field optical microscope with small sphere probe," *Science and Technology of Advanced Materials*, 8, 181–185.

[11] Pappu, R., Recht, B., Taylor, J., and Gershenfeld, N. (2002), "Physical One-Way Functions," *Science*.

[12] Photonics, N. (2012), *SuperK Extreme High Power Supercontinuum Fiber Laser Series (EXW)*, NKT Photonics.

[13] Pistol, C., Mao, V., Thusu, V., Lebeck, A. R., and Dwyer, C. (2010), "Encoded Multichromophore Response for Simultaneous Label-Free Detection," *Small*, pp. 843–850.

[14] Rivest, R., Shamir, A., and Adleman, L. (1978), "A method for obtaining digital signatures and public-key cryptosystems," *CACM*, 26, 120–126.

[15] Shannon, C. (1949), "Communication Theory of Secrecy Systems," *Bell System Technical Journal*, 4, 656–715.

[16] Shor, P. (1997), "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal of Computing 26*, pp. 1484–1509.

[17] Stinson, D. R. (2006), *Cryptography: Theory and Practice*, Chapman and Hall/CRC, 3 edn.

[18] Suen, C. Y. (1979), "n-Gram Statistics for Natural Language Understanding and Text Processing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1, 164–172.

[19] Suh, G. E. and Devadas, S. (2007), "Physical Unclonable Functions for Device Authentication and Secret Key Generation," in *Proceedings of the 44th Annual Design Automation Conference*, pp. 9–14.

[20] US National Institute of Standards and Technology (2001), "Advanced Encryption Standard (AES)," Web.

[21] Valeur, B. (2002), *Molecular Fluorescence: Principles and Applications*, Wiley-VCH.

[22] van Stokkum, I. H., Larsen, D. S., and van Grondelle, R. (2004), "Global and Target Analysis of Time-Resolved Spectra," *Biochimica Et Biophysica Acta-bioenergetics*, 1657, 82–104.

[23] Yildiz, A. and Selvin, P. R. (2005), "Fluorescence Imaging with One Nanometer Accuracy: Application to Molecular Motors," *Accounts of Chemical Research*, 38, 574–582.

[24] Zhong, H. (2010), "Photoactivated Localization Microscopy (PALM): An Optical Technique for Achieving 10-nm Resolution," *Imaging: A Laboratory Manual*.