مقاله پژوهشی

SQ-PUF: پروتکل احراز هویت مبتنی بر PUF مقاوم در برابر حملات یادگیری ماشین

سید ابوالفضل سجادی هزاوه و بیژن علیزاده

چکیده: توابع غیرهمسان فیزیکی (PUF) سخت‌افزاری را برای تولید الگویی منحصربه‌فرد از چالش- پاسخ با اهداف احراز هویت و رمزگذاری ارائه می‌دهند. یکی از ویژگی‌های مهم در این مدارها غیرقابل پیش‌بینی‌بودن است؛ به این معنی که یک مهاجم نمی‌تواند پاسخ‌های آینده را از مشاهدات قبلی پیش‌بینی کند. با این حال نشان داده شده که الگوریتم‌های یادگیری ماشین، تهدیدی قابل توجه برای PUFها هستند؛ زیرا آنها قادر به مدل‌سازی دقیق رفتار PUF می‌باشند. در این مقاله، ما تهدیدات امنیتی PUF را تحلیل و یک روش احراز هویت مبتنی بر PUF به نام SQ-PUF را ارائه می‌کنیم که می‌تواند در برابر حملات یادگیری ماشین مقاومت خوبی از خود نشان دهد. توانایی شبیه‌سازی یا پیش‌بینی آن را با مبهم‌سازی همبستگی بین جفت‌های چالش- پاسخ‌ها دشوار کردیم. نتایج تجربی نشان می‌دهند که برخلاف PUFهای موجود، حتی با مجموعه‌ای از داده‌های بزرگ هم نمی‌توان به مدل SQ-PUF حمله موفقی داشت و بیشترین دقت پیش‌بینی ٪۵۳ است که نشان‌دهنده غیرقابل پیش‌بینی‌بودن این مدل می‌باشد. علاوه بر این، یکنواختی و یکتایی در این مدل تقریباً با مقدار ایده‌آل در A-PUF یکسان باقی مانده است.

***کلیدواژه:* اینترنت اشیا، یادگیری ماشین، احراز هویت، امنیت شبکه، توابع غیرهمسان فیزیکی.**

۱- مقدمه

در عصر حاضر، پیشرفت پروتکل‌های ارتباطاتی[1] در مدارهای الکترونیکی و همچنین افزایش استفاده از اینترنت اشیا بر اهمیت بحث انتقال داده و امنیت آن در ارتباطات بین دو دستگاه افزوده است. دلایل اهمیت این موضوع را می‌توان پیشرفت روزافزون و افزایش تعداد این دستگاه‌ها برشمرد. به‌طور کلی، امنیت پروتکل‌های ارتباطی شامل دو قسمت یکپارچگی داده[2] و احراز هویت[3] است. جهت حفظ یکپارچگی، داده ارسال‌شده با استفاده از روش‌های رمزنگاری[4] امن می‌شود؛ بدین صورت که اگر مهاجم به کانال ارتباطی دسترسی پیدا کند، داده رمزنگاری‌شده نامفهومی را مشاهده می‌کند که توانایی برداشتن اطلاعات کمی از آن را دارد. در صورتی که روش رمزنگاری پیچیده‌تری ارائه شود می‌توان احتمال درزکردن اطلاعات را در هنگام حمله کاهش داد. طراحی الگوریتم رمزنگاری پیچیده، علاوه بر مباحث سنگین ریاضیات، نیاز به توان مصرفی و فضای پیاده‌سازی بیشتر و در نتیجه هزینه سخت‌افزاری بیشتری دارد؛ البته احتمال حمله موفق و خواندن اطلاعات همچنان وجود دارد. دومین چالش امنیتی در پروتکل‌های ارتباطی، بحث احراز هویت (تعیین مبدأ داده ارسال‌شده) است که زمانی مطرح می‌شود که احتمال ارسال اطلاعات توسط مهاجم وجود داشته باشد. از این رو گیرنده پیام باید توانایی تشخیص فرستنده اصلی از مهاجم را داشته باشد؛ زیرا در غیر این صورت ممکن است توسط مهاجم مورد حمله قرار گیرد.

امروزه در حوزه امنیت سخت‌افزار مفهومی با عنوان PUF[5] مطرح شده است [۱] که همانند اثر انگشت انسان برای مدارهای الکترونیکی عمل می‌کند. معماری ارائه‌شده برای PUF وابسته به تغییرات پارامترهایی همچون تأخیر مسیرها، مقادیر خازن‌ها و نظایر آن در زمان فرایند ساخت[6] مدارهای مجتمع بوده و با توجه به اینکه تغییرات فرایند ساخت به‌صورت کاملاً تصادفی تعیین می‌شوند، نحوه عملکرد PUF نیز منحصربه‌فرد و تصادفی است [۲]. به‌طور کلی می‌توان گفت که PUF یک فناوری امنیتی در حوزه سخت‌افزار است که قابل استفاده در دو مبحث مهم تولید کلید و احراز هویت بدون نیاز به ذخیره‌سازی اطلاعات خاصی در حافظه سیستم می‌باشد. از این معماری می‌توان جهت جلوگیری از نمونه‌برداری غیرمجاز مدارهای مجتمع و همچنین تولید کلید برخط و مقاوم در برابر حملات تهاجمی بهره گرفت.

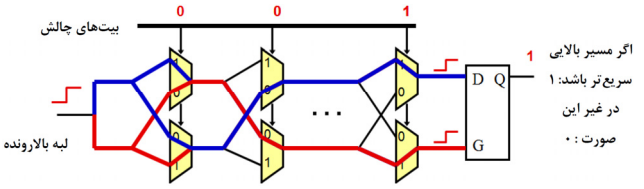
به بیان دیگر، PUF یک تابع فیزیکی غیرقابل همانندسازی است که به‌ازای یک ورودی داده‌شده، یک خروجی یکتا و تصادفی را تولید می‌کند. به هر جفت ورودی داده‌شده و خروجی یکتا و تصادفی تولیدشده، جفت چالش- پاسخ (CRP)[7] گفته می‌شود. مثال ساده‌ای از جفت چالش- پاسخ در شکل ۱ آمده که مقدار ۰۰۱ به‌عنوان چالش منجر به پاسخ ۱ خواهد شد. هر بیت چالش دو مسیر را از طریق مالتی‌پلکسرها می‌سازد که به‌صورت هم‌زمان ایجاد می‌شوند و پاسخ بر اساس مقایسه بین مسیرهای تأخیر و توسط داور مشخص می‌گردد. بدین صورت می‌توان پاسخ‌های n بیتی را با n بار تکرار این مدار و یا با استفاده از n چالش مختلف به‌دست آورد. این تابع فیزیکی برای هر سیستم الکترونیکی متناسب با ویژگی‌های فیزیکی در نظر گرفته شده، به‌صورت متفاوتی تولید گردیده و به‌دلیل ویژگی‌های غیرقابل تکرار و غیرقابل پیش‌بینی به‌عنوان یک ابزار

این مقاله در تاریخ ۲۳ بهمن ماه ۱۴۰۰ دریافت و در تاریخ ۱۸ اردیبهشت ماه ۱۴۰۲ بازنگری شد.

سید ابوالفضل سجادی هزاوه، دانشکده مهندسی برق و کامپیوتر، دانشکدگان فنی، دانشگاه تهران، تهران، ایران، (email: Sajadi@ut.ac.ir).

بیژن علیزاده (نویسنده مسئول)، دانشکده مهندسی برق و کامپیوتر، دانشکدگان فنی، دانشگاه تهران، تهران، ایران، (email: b.alizadeh@ut.ac.ir).

1. Communication Protocols
2. Data Integrity
3. Authentication
4. Cryptography Scheme
5. Physical Unclonable Function
6. Process Variation
7. Challenge Response Pair



شکل ۱: مثال چالش- پاسخ.

قوی برای احراز هویت، تشخیص اصالت و امنیت سخت‌افزاری مورد استفاده قرار می‌گیرد. مثلاً در یک سیستم تشخیص اصالت، PUF به‌عنوان یک تابع، جهت اطمینان از اینکه قطعه سخت‌افزاری اصلی استفاده شده و هر گونه جعل یا تقلبی در آن صورت نگرفته است می‌تواند به‌کار گرفته شود.

یکی از مهم‌ترین ویژگی‌های PUF، توانایی مقاومت در برابر حملات مختلف است [۳]. PUFها با استفاده از خصوصیات فیزیکی سیستم برای حفظ امنیت سخت‌افزاری در مقابل حملات فیزیکی و نفوذ به سطح سخت‌افزار مورد استفاده قرار می‌گیرند.

در این مقاله، یک روش احراز هویت مبتنی بر PUF به نام SQ-PUF ارائه خواهد شد که می‌تواند در برابر حملات یادگیری ماشین مقاومت خوبی از خود نشان دهد. این روش می‌تواند توانایی شبیه‌سازی یا پیش‌بینی خروجی PUF را با مبهم‌سازی همبستگی[1] بین جفت‌های چالش- پاسخ دشوار کند. نتایج نشان می‌دهند که با اضافه‌کردن ماژول SQ به ۸۰٪ بیت‌های ورودی (۲۵ بیت از ۳۲ بیت)، دقت حملات با روش یادگیری ماشین LR به ۵۲/۱۴٪ کاهش می‌یابد. این دقت با استفاده از مدل‌های OB-PUF و R-PUF حتی با CRPs کمتر به ترتیب ۹۵٪ و ۸۵٪ است. همچنین نتایج نشان می‌دهند که دقت حملات با روش یادگیری ماشین ANN به ۵۲/۴٪ کاهش می‌یابد. این دقت با استفاده از مدل Poly-PUF حتی با CRPs کمتر به ترتیب ۹۵٪ است.

این مقاله بدین صورت سازماندهی شده که در بخش دوم پیشینه پژوهش مورد بررسی قرار می‌گیرد. در بخش سوم پس از بررسی معماری مدل SQ-PUF پیشنهادی، مراحل ثبت نام[2] و احراز هویت را به‌صورت مرحله به مرحله شرح می‌دهیم. در بخش چهارم پس از بررسی تأثیر تعداد ماژول SQ بر مقاومت مدل، به مقایسه امنیت این مدل در مقابله با حملات پیچیده یادگیری ماشین با سایر مدل‌های مقاوم مانند PolyPUF، OB-PUF و RPUF می‌پردازیم و در انتها پس از بررسی یکتایی[3] و یکنواختی[4] پاسخ مدل پیشنهادی، نتیجه‌گیری مقاله را خواهیم دید.

۲- پیشینه تحقیق

در طی چند دهه اخیر، مطالعات بسیاری بر روی PUFها متمرکز شده‌اند و ساختارهای PUF زیادی مانند Arbiter PUF [۴] تا [۶]، RF-PUF [۷] و [۸]، Ring Oscillator (RO) PUF [۹]، MC-PUF [۱۰] و [۱۱] و همچنین پروتکل‌هایی برای احراز هویت امن مثل [۱۲] پیشنهاد شده‌اند. PUFهای ذکرشده را می‌توان به دو دسته PUFهای قوی [۵] و [۱۳] تا [۱۵] و PUFهای ضعیف [۹] و [۱۶] تا [۱۸] طبقه‌بندی کرد. PUFهای ضعیف فقط چند جفت چالش- پاسخ را تولید می‌کنند که می‌توانند به‌عنوان کلید در سیستم‌های رمزگذاری استفاده شوند؛ در صورتی که PUFهای قوی مانند Arbiter-PUF می‌توانند تعداد زیادی جفت چالش- پاسخ منحصربه‌فرد را تولید کنند. با این حال، PUFهای قوی فعلی در برابر حملات یادگیری ماشینی (ML) آسیب‌پذیر هستند [۱۹] و مهاجمان می‌توانند تعداد معینی جفت چالش- پاسخ را از کانال ارتباطی جمع‌آوری کرده تا ساختار PUF را مدل‌سازی کنند [۱۳] و [۲۰].

یک مدار Arbiter PUF با n بیت چالش و یک بیت پاسخ در شکل ۲ نشان داده شده است. وقتی چالش روی ورودی قرار می‌گیرد پاسخ در هر مرحله از دو مسیر منتقل می‌شود. چالش‌های ورودی تعیین می‌کنند که مالتی‌پلکسرها کدام مسیر را برای انتقال به خروجی انتخاب کنند. در انتها یک داور مانند یک فلیپ‌فلاپ نوع D با توجه به زمان رسیدن سیگنال هر دو مسیر، وضعیت خروجی را انتخاب می‌کند؛ به این معنی که اگر سیگنال منتقل‌شده زودتر از لبه بالارونده کلاک به پایه D برسد، پاسخ یک و در غیر این صورت صفر است.

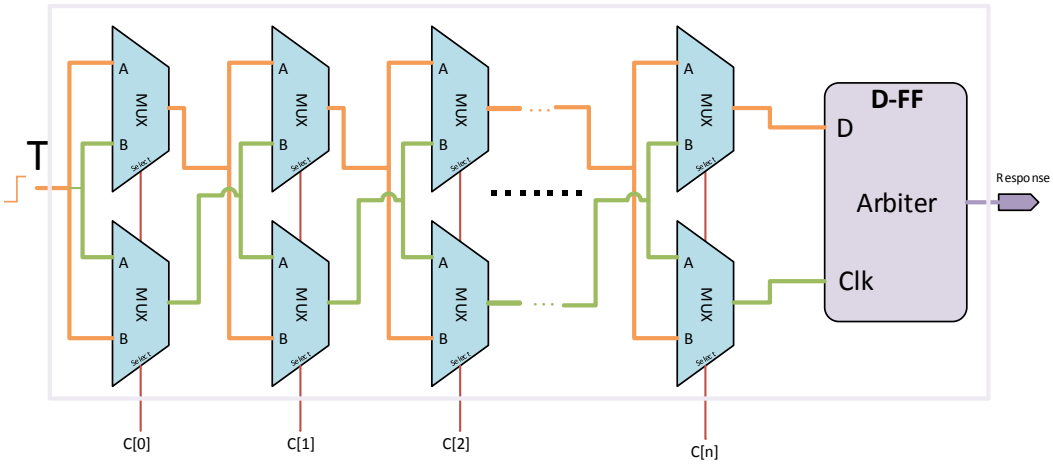
از آنجا که تأخیر هر کدام از مسیرها و مالتی‌پلکسر، تحت تأثیر تغییرات مرحله ساخت می‌باشد، پیش‌بینی آنها قبل از تولید و مدل‌سازی آنها پس از تولید، کار بسیار سختی است. اما از آنجا که بخش‌های مختلف مالتی‌پلکسر برای چالش‌های مختلف اشتراک‌گذاری می‌شوند، پس بین جفت چالش پاسخ‌های مختلف همبستگی وجود دارد و این یک موقعیت بسیار خوب برای حمله‌کننده است که به وسیله یادگرفتن این همبستگی‌ها مقادیر را پیش‌بینی کنند.

نویسندگان [۲۱]، جامع‌ترین الگوریتم‌های یادگیری ماشین (ML) را برای مدل‌سازی و حمله به PUFها بررسی کردند و نیز توانسته‌اند تا با کمک روش رگرسیون لجستیک (LR) و $۳۹/۲\times۱۰^{۳}$ جفت چالش- پاسخ در ۱۰/۲ ثانیه به دقت ۹۹٪ از مدل‌سازی Arbiter PUF برسند. همچنین XOR-APUF با ۵ مرحله را با ۵۰۰×۱۰^{۲} جفت چالش- پاسخ بعد از گذشت ۱۹:۳۶ ساعت مدل‌سازی کردند. از سوی دیگر، نویسندگان در [۲۲] چارچوب جدیدی را برای بهبود سرعت حمله به PUFها با استفاده از تکنیک‌های یادگیری فعال ارائه کردند و نشان دادند که یادگیری فعال به‌طور قابل توجهی می‌تواند سرعت حمله به PUFها را بهبود بخشد و نیز با بهره‌گیری از روش ماشین بردار پشتیبان (SVM) و یادگیری فعال، فقط با استفاده از ۲۰۰ جفت چالش- پاسخ می‌توان به یک MUX-PUF با ۶۴ مرحله مالتی‌پلکسر حمله موفق داشت. همچنین نتایج تجربی نشان داد که برای خطای پیش‌بینی زیر ۴٪ به ۲۷۹۰ جفت چالش- پاسخ نیاز است که با روش یادگیری فعال به ۸۱۱ جفت چالش- پاسخ کاهش می‌یابد. در واقع، بیشتر مدارهای PUF موجود با استفاده از تکنیک‌های یادگیری ماشین، مستعد مدل‌سازی حمله هستند. در [۲۳] امنیت ۲۱ پروتکل احراز هویت مبتنی بر PUF تجزیه و تحلیل گردید و مشکلات متعدد آنها نشان داده شد. همچنین نویسنده در [۲۴]، در مورد توسعه حملات جعل هویت کارآمد در پنج پروتکل قوی احراز هویت مبتنی بر PUF Arbiter تحقیق کرد.

در طول دهه گذشته، ساختارهای مختلفی از PUFها ارائه شده است که همه آنها به دنبال بهبود امنیت در بحث احراز هویت هستند. در ادامه، برخی از ایمن‌ترین مدل‌های PUF از جمله Slender PUF [۲۵]، Poly-PUF [۲۶]، R-PUF [۲۷] و OB-PUF [۲۸] مورد بررسی قرار گرفته‌اند و سپس یک مدل جدید برای حل امنیت مطرح شده است.

یکی از گونه‌های PUF که در [۲۵] معرفی شد، slender PUF نام دارد که از بیت‌های تصادفی استفاده می‌کند. به بیان دیگر، اگر در یک Arbiter PUF تعداد m بیت پاسخ وجود داشته باشد، خروجی را می‌توان

1. Correlation
2. Enrollment
3. Uniqueness
4. Uniformity



شکل ۲: مدار Arbiter PUF.

به‌صورت $[r_1, r_2, \ldots, r_m]$ نمایش داد که این خروجی در slender PUF به‌صورت $[t_1, \ldots, t_k, r_1, r_2, \ldots, r_m, t_{k+1}, \ldots, t_m]$ تغییر می‌کند. مقادیر t_1 تا t_m تصادفی انتخاب می‌شوند. حال وقتی یک حمله‌کننده تعداد $2 \times m$ بیت از پاسخ را به‌دست می‌آورد، نمی‌تواند بفهمد که کدام یک از بیت‌ها مربوط به پاسخ اصلی هستند؛ اما به هر حال [۲۹] با استفاده از یک استراتژی تکاملی، موفق به شکستن امنیت این PUF شده است. بنابراین این اقدامات برای جلوگیری از حملات مدل‌سازی یا حملات مبتنی بر یادگیری ماشین کافی نبوده است.

یکی از پروتکل‌های مطرح‌شده برای مقابله با حملات مدل‌سازی، PolyPUF می‌باشد [۲۶] که برای مقابله با حمله توسط یک TRNG[1]، عددی را به‌صورت تصادفی تولید و با چالش ورودی XOR می‌کند. پس از محاسبه پاسخ توسط n مدل مختلف A-PUF (که n در اینجا تعداد بیت پاسخ است)، دوباره تمام پاسخ‌ها با مقدار تصادفی جدید تولیدشده، XOR می‌شوند. تعداد بیت پیشنهادی برای عدد تصادفی ورودی ۲ و خروجی ۳ بیت و طول داده چالش پیشنهادی ۳۲ و ۶۴ بیت است؛ اما نویسندگان درباره آنکه ۳ مضربی از ۳۲ یا ۶۴ نیست، اظهار نظری نکردند.

به هر حال نویسندگان در [۲۴] با استفاده از دسته‌بندی داده‌ها توانستند به این مدل، حمله موفقی داشته باشند. به این صورت که ادعا شده اگر فاصله همینگ بین دو چالش متوالی یک باشد، در جایی که فاصله همینگ دو خروجی متناظر کمترین شود، نشانه این است که مقدار عدد تصادفی که با چالش و پاسخ XOR شده بین هر دو چالش ثابت مانده است. بدین صورت یک لیست از داده‌ها تشکیل و توسط یک شبکه با یک نورون تکی موفق به حمله با دقت بالای ۹۰٪ به این مدل شده است. یکی دیگر از مدل‌های مقاوم ارائه‌شده با نام OBPUF شناخته می‌شود که در [۲۸] معرفی گردیده و به‌وسیله بخش کنترل‌کننده چالش، مقدار چالش ورودی را تغییر می‌دهد. هر بار به‌صورت تصادفی یکی از الگوها توسط بخش فوق انتخاب می‌گردد و به ورودی اعمال می‌شود که چالش ورودی به‌صورت $[0,1,0,1,0,C_6,C_7,\ldots,C_{64}]$ یا $[C_1,C_2,\ldots,C_{59},1,0,1,0,1]$ تغییر می‌کند. نتایج نشان داده که پس از جمع‌آوری پاسخ‌ها و حمله توسط الگوریتم رگرسیون لجستیک با 10^6 داده نتوانستند به دقت بیشتر از ۷۲٪ دست پیدا کنند. البته این احتمال بسیار زیاد است که Arbiter PUF مورد نظر مقادیر نامی یکسانی را برای همه پاسخ‌های مربوط به یک چالش خاص داشته باشد. مرجع [۲۴] به وسیله جداکردن این مقادیر و آموزش یک مدل با الگوریتم رگرسیون لجستیک به دقت ۸۵٪ رسید. در مرحله بعدی با استفاده از این مدل به عنوان یک ابزار ابهام‌زدا اقدام به جمع‌آوری داده‌های بیشتر با فاصله همینگ کم نسبت به خروجی پیش‌بینی‌شده نمودند و به دقت بالای ۹۵٪ دست یافتند.

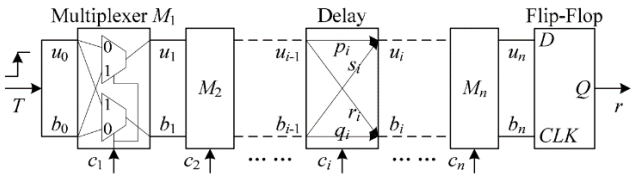
مدل RPUF [۲۷] یکی از امن‌ترین مدل‌های ارائه‌شده است که دو حالت کاری دارد. برای جلوگیری از حملات یادگیری ماشین، چالش دریافت‌شده را به‌صورت تصادفی معکوس می‌کند. در حالت کاری اول، تمام بیت‌های چالش دریافتی را معکوس می‌کند و یا بدون تغییر، انتقال می‌دهد که این عمل توسط یک TRNG انتخاب می‌شود. در حالت کاری دوم برای دستیابی به امنیت بیشتر در برابر حملات، TRNG به‌جای یک بیت تصادفی دو بیت تولید می‌کند که این دو بیت می‌تواند ۴ عملکرد مختلف داشته باشد: در حالت اول، چالش دریافت‌شده را بدون تغییر به خروجی انتقال می‌دهد. در حالت دوم، چالش دریافت‌شده را کاملاً عکس می‌کند و در دو حالت بعدی، بیت‌های نیمه بالایی یا نیمه پایینی چالش دریافت‌شده را عکس می‌کند. نویسندگان ادعا کردند که مدل ارائه‌شده نمی‌تواند با دقت بیشتر از حدود ۷۵٪ مورد حمله قرار گیرد. البته در [۲۴] توسط یک استراتژی یادگیری جایگزین توانستند به دقت ۹۰٪ برسند؛ اما حملات دیگری نیز در [۳۰] روی این مدل انجام شده است.

اکثر مدل‌های پیشنهادی دارای ضعف‌هایی در روند پنهان‌سازی رابطه بین چالش و پاسخ هستند. در این مقاله سعی شده که با وابسته‌سازی چالش‌ها به زمان و یکدیگر و همچنین اضافه‌کردن یک خاصیت تصادفی به آنها با قابلیت تشخیص جواب صحیح برای احراز هویت، به هدف افزایش مقاومت مدل در برابر حملات یادگیری ماشین دست پیدا کنیم. مهم‌ترین فرض مسأله این است که حمله‌کننده، اطلاعی از ساختار داخلی PUF ندارد.

۳- روند پیشنهادی احراز هویت مبتنی بر PUF (SQ-PUF)

هدف از ارائه مدل SQ-PUF، جلوگیری از استخراج رابطه همبستگی بین چالش و پاسخ توسط روش‌های یادگیری ماشین و مقاوم‌سازی مدل در برابر حملات مدل‌سازی می‌باشد؛ بدین صورت که پس از مقاوم‌سازی نیز سرور بتواند با استفاده از یک روش بازیابی، عملیات احراز هویت را انجام دهد. همبستگی بین چالش و پاسخ به معنی همبستگی بین خصوصیات فیزیکی خاص PUF و پاسخ تولیدشده به چالش خاص است. توجه شود که تعداد مالتی‌پلکسرها در مسیر، پارامترهای ترانزیستوری و

1. True Random Number Generator



شکل ۳: تأخیر مسیرهای مالتی‌پلکسرها در Arbiter-PUF [۲۷].

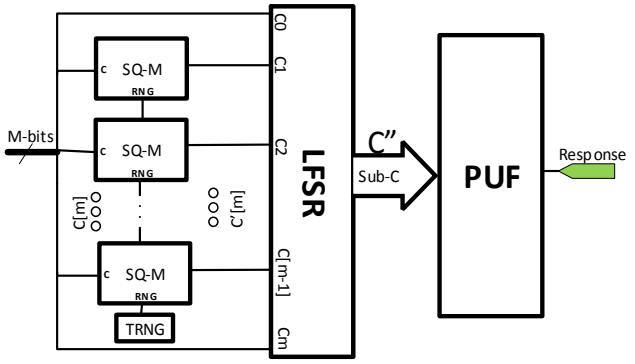
سایر تغییرات در فرایند ساخت، همه بر پاسخ تولیدشده توسط PUF تأثیر دارند و بنابراین ارتباط نزدیک دو چالش با یکدیگر می‌تواند به ارتباط نزدیک دو پاسخ منجر شود. مثلاً در نظر بگیرید که نمونه ارائه‌شده در شکل ۱ از روی چالش ۰۰۱ پاسخ ۱ را تولید کرده است. به کمک چهار مدار متوالی مشابه، پاسخ چهاربیتی به چالش ۰۰۱ تولید خواهد شد که برابر با ۱۱۰۱ است. حال اگر با چالش ۱۰۱ پاسخ ۱۰۰۱ و با چالش ۱۱۱ پاسخ ۰۱۰۱ تولید شوند، احتمالاً اولین بیت چالش با دو بیت اول پاسخ، همبستگی دارد. توجه شود که وضعیت وقتی بدتر خواهد شد که تعداد بیت‌های پاسخ را افزایش دهیم؛ به عبارت دیگر، اگر از n مدار متوالی برای تولید n بیت پاسخ استفاده شود، همبستگی بسیار بیشتر خواهد شد.

برخلاف Arbiter PUF که فقط یک بیت برای پاسخ دارد، در بسیاری از کاربردها به چندین بیت پاسخ نیاز است. برای این مسئله دو راه حل وجود دارد [۹] و [۱۴]. اولین راه این است که مانند مدل‌هایی مثل OBPUF یا PolyPUF از تعداد n عدد Arbiter PUF موازی برای تولید n عدد پاسخ مختلف استفاده کنیم. اما با توجه به اینکه بسیاری از کاربردها مانند احراز هویت در سیستم‌های اینترنت اشیا (IoT) نیازمند یک سخت‌افزار با منابع کم است، از این رو در این مقاله با استفاده از روش دوم یعنی استفاده از یک LFSR[1] و یک Arbiter PUF طراحی خود را انجام دادیم. در این روش با استفاده از یک LFSR از چالش ورودی به تعداد بیت مورد نیاز پاسخ، زیرچالش را تولید و به A-PUF اعمال می‌کنیم.

همان طور که در شکل ۲ توضیح داده شد، یک Arbiter-PUF بر اساس بیت‌های چالش مشخص‌شده، مسیرهای مختلفی از MUXها را طی کرده و در نتیجه پاسخ‌های مختلفی ایجاد خواهد کرد. اگر مسیری که به پایه D فلیپ‌فلاپ ختم می‌شود زودتر برسد، داور (فلیپ‌فلاپ) خروجی را یک کرده و در غیر این صورت خروجی صفر می‌شود. این کار همبستگی زیادی ایجاد می‌کند که توسط الگوریتم‌های یادگیری ماشین قابل استفاده در جهت آموزش یک شبکه عصبی است.

در [۲۷]، نحوه استفاده از این همبستگی به‌منظور حمله به مدل PUF تشریح گردید. شکل ۳ همان Arbiter-PUF را نشان می‌دهد که در آن اطلاعات تأخیر مسیرهای مالتی‌پلکسر M_i با متغیرهای r_i، s_i، q_i و p_i مشخص شده‌اند. به‌ازای یک چالش معین، پاسخ r از طریق (۱) محاسبه خواهد شد

$$r = \theta(\vec{W}\times\vec{\Phi}) \rightarrow \theta(x) = \begin{cases} ۰ & , \ x < ۰ \\ ۱ & , \ x \geq ۰ \end{cases}$$
$$\vec{W} = (w_۱,\ldots,w_{n+۱}) \rightarrow w_i = \frac{\delta_{i-۱}^{:} + \delta_{i-۱}^{۱} + \delta_i^{:} - \delta_i^{۱}}{۲} \qquad (۱)$$
$$\vec{\Phi} = (\phi_۱,\ldots,\phi_n,۱) \rightarrow \phi_l = \prod_{i=l}^{n}(۱-۲c_i)$$
$$, \ \delta_i^{:} = q_i - p_i \ , \ \delta_i^{۱} = r_i - s_i$$



شکل ۴: معماری کلی SQ-PUF.

در این رابطه فرض کنید که برای یک جفت چالش- پاسخ معین، مقدار $\vec{\Phi}$ و r را می‌دانیم. با استفاده از رابطه $\vec{W}\times\vec{\Phi} = ۰$، بردار $\vec{W}$ می‌تواند یک ابرصفحه[2] جداکننده را تعیین کند که چالش‌هایی را که پاسخ آنها صفر می‌شوند از دیگر چالش‌ها جدا سازد. بدین ترتیب روش‌های یادگیری ماشین، پس از یافتن این ابرصفحه می‌توانند چالش‌ها را دسته‌بندی نمایند. از این رو در روش‌های مختلف جهت مقاوم‌سازی PUFها سعی شده که ارتباط و همبستگی بین چالش- پاسخ را از بین ببرند. در روش‌هایی مثل XOR-PUF پاسخ را مخفی و در روش‌هایی مثل RPUF و OB-PUF چالش را مخفی ساختند؛ اما به هر حال توسط استراتژی‌هایی که قبلاً ذکر شد، برخی توانستند به این ابرصفحه دست یابند.

هدف ما در این مقاله، ارائه نوع دیگری از PUFهاست که با استفاده از وابستگی زمانی، ارتباط بین چالش و پاسخ را مخفی کنیم. به عبارت دیگر هدف، جلوگیری از یادگرفتن ارتباط بین چالش‌ها و پاسخ‌ها و پنهان‌کردن ابرصفحه از روش‌های یادگیری ماشین است. برای این منظور باید فاصله بین چالش‌ها را زیاد و از چالشی استفاده کنیم که در داخل PUF به وجود آمده است. همچنین با وابسته‌کردن چالش‌ها به زمان می‌توان از پی‌بردن حمله‌کننده به آنها جلوگیری نمود.

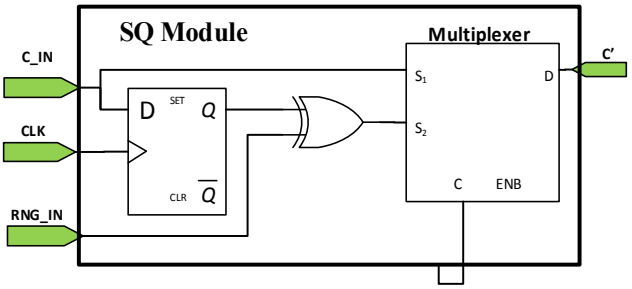
در شکل ۴ یک مدل کلی از SQ-PUF با m بیت چالش نشان داده شده است. کلمه SQ از Sequence یا توالی گرفته شده که به معنی ایجاد یک توالی بین چالش‌های ورودی است. این ماژول‌ها را می‌توان به همه (۱۰۰٪) یا فقط به بخشی از چالش‌ها تخصیص داد. پس از تغییر چالش‌ها به‌وسیله ماژول‌های SQ، آنها به یک LFSR وارد گردیده تا زیرمجموعه‌هایی از این چالش برای تولید بیت‌های مختلف پاسخ تولید شوند.

۳-۱ ماژول SQ

همان طور که قبلاً اشاره شد می‌توان با دو چالش نزدیک به هم، دو پاسخ نزدیک به هم دریافت کرد. حمله‌کننده می‌تواند از این همبستگی، استفاده و با یک شبکه عصبی، تابع داخلی PUF را بازیابی کند. برای جلوگیری از این اتفاق باید فاصله بین چالش‌ها را زیاد کرد. همچنین با وابسته‌کردن چالش‌ها به زمان می‌توان از پی‌بردن حمله‌کننده به آنها جلوگیری نمود. ایده اصلی SQ-PUF، وابسته‌کردن دو چالش متوالی به یکدیگر است؛ به‌عبارت دیگر در این مدل PUF، پاسخ متأثر از دو ویژگی است که شامل ترتیب واردشدن چالش‌ها و عدد تصادفی تولیدشده توسط RNG می‌باشد. این ماژول می‌تواند به همه و یا فقط به برخی از چالش‌ها اعمال شود. این بیت‌ها در زمان ساخت PUF به‌صورت تصادفی انتخاب

1. Linear-Feedback Shift Register

2. Hyper Plane



شکل ۵: ماژول SQ.

می‌شوند. به‌صرف وجودداشتن یک RNG یک‌بیتی، این مدار خاصیت تصادفی‌بودن خود را حفظ می‌کند و از این رو پاسخ هر چالش با توجه به چالش قبلی دارای دو جواب صحیح است. از آنجا که قبلاً کارهای زیادی برای طراحی RNG انجام گردیده [۳۱] تا [۳۳] و هدف ما طراحی یک مدل مقاوم در برابر حملات مدل‌سازی است، پس فرض می‌کنیم که یک RNG شایسته، بهینه و با پاسخ‌دهی دقیقاً پنجاه‌پنجاه در این مدل استفاده شده است.

در ماژول SQ که در شکل ۵ آمده، با استفاده از یک D فلیپ‌فلاپ چالش را به حالت قبلی وابسته می‌کنیم و حالت اولیه را می‌توان صفر در نظر گرفت. در این بخش از یک XOR جهت تصادفی‌سازی چالش بهره می‌بریم. اگر عدد تصادفی تولیدشده برابر یک باشد، چالش عکس می‌شود و در غیر این صورت بدون تغییر به خروجی انتقال می‌یابد. همچنین در این ماژول به کمک یک مالتی‌پلکسر که فقط برای بار اول به آن دسترسی داریم (می‌توان آن را به صورت یک فیوز یک بار مصرف در نظر گرفت)، ارتباط مستقیمی بین چالش ورودی و PUF برقرار می‌شود؛ یعنی به‌واسطه این مالتی‌پلکسر می‌توان با نادیده‌گرفتن ماژول‌های SQ به هسته PUF دسترسی مستقیم داشت که به‌منظور آموزش سرور تعیین شده است.

۳-۲ مرحله ثبت نام

مراحل بخش ثبت نام فقط یک بار جهت شناسایی تابع داخلی PUF مورد نظر به سرور برای انجام احراز هویت انجام می‌شود. البته با توجه به موضوع سالخوردگی در وسایل الکترونیکی و تأثیر آن بر روی پاسخ‌های تولیدشده، این مرحله می‌تواند هر چند وقت یک بار انجام شود [۳۴]. در این قسمت، سرور می‌تواند به بخش‌های خاصی نظیر مالتی‌پلکسرهای داخلی دسترسی داشته باشد؛ اما پس از اتمام کار، این دسترسی‌ها از بین می‌روند که این عمل می‌تواند به صورت یک فیوز محقق شود. ابتدا سرور، تعداد N چالش را به‌صورت تصادفی تولید نموده و به PUF ارسال می‌کند که تعداد این چالش‌ها باید به اندازه کافی بزرگ باشند. در این فاز، SQ-PUF توسط مالتی‌پلکسرهای موجود در ماژول SQ، دسترسی مستقیم چالش‌ها به Arbiter PUF را فراهم می‌کند. سپس به‌ازای هر چالش واردشده برای تولید α بیت پاسخ، α بار زیرمجموعه‌های این چالش توسط LFSR تولید و به Arbiter PUF اعمال می‌شود. پاسخ‌های تولیدشده به سرور ارسال می‌گردند. بدین صورت در سرور، دسته‌ای از چالش- پاسخ‌های معین را داریم که مستقیماً توسط Arbiter PUF تولید شده‌اند. پس سرور می‌تواند توسط این دسته، مدلی را آموزش دهد تا این مدل، ابرصفحه جداکننده پاسخ‌ها را به‌ازای چالش‌ها با دقت بالایی پیدا کند و به‌عبارت دیگر، وابستگی بین چالش- پاسخ را با دقت بالایی آموزش ببیند.

Algorithm 1 Authentication (∞ times)

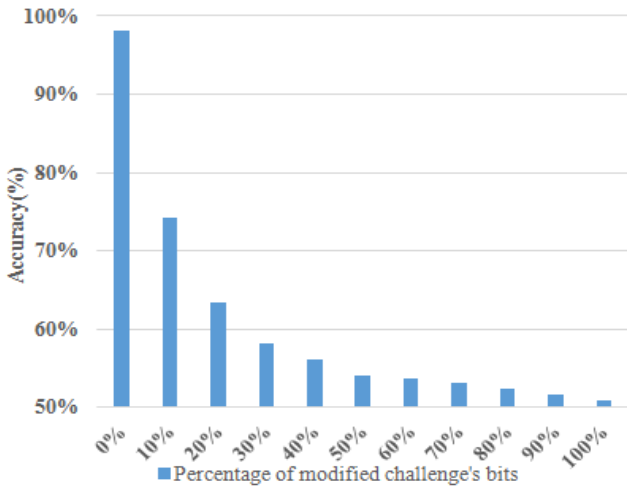
1: ▷ $\leftarrow Random-number-Generation(m)$
2: $C \leftarrow Server$
3: n = TRNG()
4: $C' = D_{FF}(C)$
5: $C'' = C' \mathcal{XOR}$ n
6: $C''_{(1,2,\ldots,\alpha)} = LFSR(C'')$
7: **for** $j = 1, 2, \ldots, \alpha$ **do**
8: $\quad \mathbf{r}_{(j)} = \text{ArbiterPUF}(\mathcal{C}''_j)$
9: **end for**
10: $\mathbf{r}_{(j)} \rightarrow Server$
▷ $Server:$
11: Recive $\mathbf{r} \leftarrow PUF$
12: **if** $Firstsession$ **then** $C_{old} \leftarrow Save\ Challenge$ ▷ $NotValid$
13: **else**
14: $\quad C' = Sim(D_{FF}(C))$
15: $\quad$ **for** $\mathbf{n} = 0, 1$ **do**
16: $\quad\quad C''_{\mathbf{n}} = C' \mathcal{XOR}\ \mathbf{n}$
17: $\quad\quad r_{\mathbf{n}} = Predict(C''_{\mathbf{n}})$
18: $\quad$ **end for**
19: $\quad h = \min(\mathbf{HD}(\mathbf{r}, r_0), \mathbf{HD}(\mathbf{r}, r_1))$
20: $\quad$ **if** $h > \epsilon$ **then** $Reject$
21: $\quad$ **else**
22: $\quad\quad C_{old} \leftarrow Save\ Challenge$

شکل ۶: الگوریتم مرحله احراز هویت SQ-PUF.

۳-۳ مرحله احراز هویت

در الگوریتم شکل ۶، مراحل بخش احراز هویت نشان داده شده است. هر بار که سرور نیاز به احراز هویت داشته باشد با طی‌کردن این مراحل می‌تواند روند احراز هویت را به‌خوبی انجام دهد. همچنین این مراحل نشان‌دهنده آن است که یک روند بازیابی قابل اجرا برای مدل SQ-PUF ارائه شده است. در این بخش هیچ دسترسی به مالتی‌پلکسرهای داخل ماژول SQ وجود ندارد. در ابتدای یک نشست جهت انجام عملیات احراز هویت دستگاه، سرور m بیت تصادفی را به‌عنوان یک چالش تولید و به دستگاه ارسال می‌کند (خط ۱) و در دستگاه یک عدد تک‌بیتی تصادفی تولید می‌شود (خط ۳). همان طور که قبلاً اشاره شد در مدل SQ-PUF، ماژول‌های SQ می‌توانند فقط به برخی بیت‌های چالش اعمال شوند؛ بنابراین فقط بیت‌هایی از چالش ورودی که این ماژول به آنها اعمال شده، تغییر می‌کنند. این تغییر در دو مرحله اتفاق می‌افتد: در ابتدا توسط فیلیپ‌فلاپ نوع D، وابستگی به چالش قبلی انجام می‌شود (خط ۴) و سپس خاصیت تصادفی‌بودن توسط گیت XOR و بیت تصادفی تولیدشده در مرحله قبل به آن اضافه خواهد شد (خط ۵). خروجی "C که چالش تغییریافته است به‌عنوان حالت اولیه به LFSR اعمال می‌شود (خط ۶). پس از این مرحله به تعداد بیت‌های مورد نیاز پاسخ (α بیت) زیرمجموعه از چالش توسط LFSR تولید و به A-PUF داده می‌شود (خط ۸) و پاسخ تولیدشده به سرور ارسال می‌گردد (خط ۱۰). استفاده از LFSR باعث کاهش منابع مصرفی در مدار می‌شود؛ اما برای تولید α بیت پاسخ احتیاج به α سیکل کلاک داریم. در مواردی که محدودیت منابع داریم، این مدت زمان قابل پذیرش است.

در بخش سرور، اگر ماژول‌های SQ در SQ-PUF مورد نظر فقط به برخی بیت‌های چالش ورودی اعمال شده باشند، اینجا هم دقیقاً باید روند تغییر چالش ورودی در دستگاه شبیه‌سازی شده و در سرور دقیقاً همان بیت‌های چالش تغییر داده شوند (خط ۱۴). مثلاً ۸۰٪ بیت‌ها برای بار اول در مرحله ثبت نام به‌صورت تصادفی انتخاب شده و ماژول‌های SQ سر راه آنها می‌توانند قرار گیرند. پس از تغییر چالش، آن را با ۰ و ۱، XOR می‌کنیم (خط ۱۶). سپس پاسخ هر دو خروجی تولیدشده توسط مدلی که در فاز ثبت نام آموزش داده شده بود، پیش‌بینی می‌شوند (خط ۱۷) و در



شکل ۷: بررسی تأثیر تعداد ماژول SQ بر مقاومت مدل در برابر حملات.

انتها حتماً باید پاسخ دریافت‌شده از SQ-PUF حداقل معادل یکی از پاسخ‌های پیش‌بینی‌شده باشد. این مرحله را با محاسبه فاصله همینگ بین آنها بررسی می‌کنیم (خطوط ۱۹ و ۲۰). پس از انجام روند احراز هویت، چالش تولیدی به عنوان C_{old} در سرور ذخیره می‌شود (خط ۲۲) تا در نشست بعدی جهت احراز هویت از آن استفاده گردد. در خط ۲۰، ε برابر صفر یا مقداری کم است که با توجه به نویز موجود در محیط می‌توان آن را محاسبه نمود. لازم به ذکر است که با توجه به وابستگی خروجی SQ-PUF به ترتیب چالش‌ها، اولین پاسخ را غیرمعتبر در نظر می‌گیریم (خط ۱۲) و از آن به بعد احراز هویت را انجام می‌دهیم. طبق این روند، وابستگی بین چالش- پاسخ تا حد زیادی از بین رفته و ارتباط بین آنها غیرقابل پیش‌بینی می‌شود.

٤- نتایج

برای انجام شبیه‌سازی‌ها نیاز به پیداکردن محدوده مقدار تأخیر مسیرها در PUF است. یکی از روش‌های موجود برای بررسی تغییرات ساخت، استفاده از شبیه‌سازی‌های پیاپی به‌صورت تصادفی با روش مونت‌کارلو در محدوده گوشه‌های مدار می‌باشد؛ لذا ابتدا یک Arbiter PUF با ۶۴ چالش ورودی و یک بیت پاسخ در نرم‌افزار Hspice پیاده‌سازی گردید. سپس به کمک الگوریتم مونت‌کارلو (در ۱۰۰۰ شبیه‌سازی)، این مدار از نظر تأثیر فرایند ساخت بر تأخیر مسیرهای مختلف مورد بررسی قرار گرفت که در بدترین حالت، مقادیر میانگین ۳/۵۲ پیکوثانیه و انحراف معیار ۱۱ پیکوثانیه به‌دست آمدند. در مدل‌سازی‌ها مقادیر تأخیر مسیرها با استفاده از یک توزیع نرمال در این محدوده تعیین شدند تا نزدیک‌ترین حالت به واقعیت شبیه‌سازی شود.

٤-١ بررسی تأثیر تعداد ماژول SQ

نمودار ارائه‌شده در شکل ۷، تأثیر میزان استفاده از ماژول SQ در بیت‌های چالش را بر مقاومت مدل در برابر حملات یادگیری ماشین بیان می‌کند. این حملات توسط یک مدل ANN با چهار لایه به ترتیب ۲۰۰، ۱۰۰، ۸۰ و ۶۴ نورون انجام شده است. در لایه‌های مخفی این شبکه از تابع فعال‌ساز ReLu استفاده گردیده و لایه آخر دارای تابع فعال‌ساز Sigmoid می‌باشد که به‌وسیله دیتاستی به اندازه 5×10^5 آموزش داده شده و با این حجم از داده می‌توان گفت که مقاومت کامل این مدل مورد بررسی قرار گرفته است. این نمودار نشان می‌دهد که با افزایش تعداد ماژول‌های SQ، همبستگی مدل PUF کمتر شده و در نتیجه شبکه عصبی نمی‌تواند آن را مدل‌سازی کند و رفتار سیستم را تشخیص دهد. این نمودار نشان می‌دهد در صورتی که هیچ ماژول SQ به PUF اضافه نشود، شبکه عصبی می‌تواند تا حدود ۹۸/۱۶٪ دقت به مدل حمله کند. به محض اضافه‌کردن ماژول SQ به ۱۰٪ بیت‌های ورودی (در اینجا یعنی ۳ بیت از ۳۲ بیت)، دقت این حمله تا ۷۴/۱۶٪ کاهش یافته است؛ یعنی با اضافه‌کردن تنها سه ماژول SQ به مدل PUF توانستیم مقاومت مدل را حدود ۲۶٪ افزایش دهیم. همان طور که در شکل ۶ مشخص است با اضافه‌کردن ماژول SQ به ۸۰٪ بیت‌های ورودی (۲۵ بیت از ۳۲ بیت)، مدل PUF به مقاومت ۶۸/۴۵٪ می‌رسد که مقاومت خوبی به حساب می‌آید. پس در ادامه، شبیه‌سازی‌های خود را توسط مدل SQ-PUF با ۸۰٪ ماژول SQ انجام می‌دهیم.

جدول ۱: بررسی مقاومت SQ-PUF (۸۰٪ ماژول SQ) در برابر حملات.

	CRPs	LR	ANN
Poly-PUF [۲۶]	10^5	[۲۶] ۵۱/۹۷٪	[۲۴] ~۹۵%*
OB-PUF [۲۸]	2×10^5	[۲۴] ~۹۵٪	–
R-PUF (L۲) [۲۷]	10^3	[۲۴] ~۸۵٪	–
SQ-PUF	4×10^5	۵۲/۱۴٪	۵۲/۴٪

* A pair of single-neuron networks

٤-٢ بررسی مقاومت SQ-PUF در برابر حملات

جدول ۱ بیانگر مقاومت مدل SQ-PUF با ۸۰٪ ماژول SQ در برابر دو روش مختلف از حملات پراستفاده در مقایسه با سایر مدل‌های PUF است. همان طور که مشخص می‌باشد، مدل پیشنهادی در برابر این گونه از حملات مقاوم بوده و همچنین لازم به ذکر است که SQ-PUF توسط دیتاستی به اندازه 4×10^5 در تمامی روش‌ها مورد حمله قرار گرفته و نیز برای بررسی بیشتر، مدل پیشنهادی با روش SVM مورد ارزیابی قرار گرفت که دقت حمله در این روش هم از ۲۲/۵۰٪ فراتر نرفت. نتایج این جدول نشان می‌دهند که حمله به روش LR در مدل‌های OB-PUF [۲۱] و R-PUF [۲۷] با درصد بالایی موفق بوده است؛ اما این حمله در مدل پیشنهادی ما حتی با CRPs دو تا چهارصد برابر نیز نتوانسته موفق باشد. از طرف دیگر، اگرچه حمله به روش ANN در مدل Poly-PUF موفق بوده است با CRPs چهاربرابری در مدل پیشنهادی ما نتوانسته که موفق عمل کند. دلیل مقاومت بالای مدل پیشنهادی آن است که حملات مبتنی بر یادگیری ماشین قادر نیستند رابطه نزدیکی بین چالش‌ها پیدا کنند. این اتفاق به کمک ماژول‌های SQ پیشنهادی رقم خورده که باعث تغییر چالش ورودی می‌شوند. توجه گردد که این تغییر، علاوه بر پاسخ قبلی بر اثر عدد تصادفی نیز ایجاد شده و بنابراین با استفاده از تغییر با زمان و تصادفی‌کردن چالش‌ها توانستیم رابطه چالش‌ها و پاسخ‌ها را پنهان نماییم.

٤-٣ آنالیز یکنواختی در SQ-PUF

یکنواختی عبارت است از تناسب وجود صفرها و یک‌ها در پاسخ‌های حاصل از PUF مورد نظر که ایده‌آل این معیار برابر ۵۰٪ می‌باشد و یکی از مهم‌ترین معیارها برای طراحی یک مدل جدید PUF است و تعیین می‌کند که آیا صفر و یک‌ها بین پاسخ‌ها به‌صورت یکنواخت توزیع شده است یا خیر.

جدول ۲ نتایج این آنالیز را نشان می‌دهد. برای محاسبه این مقدار برای مدل اصلی Arbiter PUF و مدل ارائه‌شده SQ-PUF از تعداد 2×10^4

جدول ۲: بررسی یکنواختی و یکتایی مدل SQ-PUF.

	تعداد بیت‌های چالش	تعداد بیت‌های پاسخ	تعداد CRPها	یکنواختی (٪)	یکتایی (٪)
PUF Arbiter	۶۴	۶۴	۲×۱۰^{۴}	۵۰٫۵۴۷	۴۹٫۱۹۷
RPUF	۶۴	۶۴	۱×۱۰^{۴}	۵۱٫۱	۴۹٫۷
SQ-PUF	۶۴	۶۴	۲×۱۰^{۴}	۵۰٫۵۶۱	۴۹٫۱۰۱

جدول ۳: سربار سخت‌افزاری SQ-PUF در مقایسه با PUF متداول.

	C (۶۴)−R (۶۴)	LUTs	FFs
	Basic-PUF	۸۱۹۲	۶۴
	LFSR	۳۴	۶۴
SQ-PUF	PUF − ۶۴ − ۱bit	۱۲۸	۱
	SQ-Module	۱	۱

چالش- پاسخ مختلف استفاده کردیم. همچنین درصد گزارش‌شده در جدول ۲ بر اساس تعداد یک‌ها نسبت به‌کل پاسخ‌ها می‌باشد و همان طور که از نتایج نیز مشخص است، ماژول SQ تأثیر خاصی روی یکنواختی مقادیر نمی‌گذارد.

٤-٤ آنالیز یکتایی در SQ-PUF

یکتایی عبارت است از ارزیابی تفاوت پاسخ‌های تولیدشده توسط PUFهای مختلف با اعمال یک چالش مشابه که یکی دیگر از معیارهای مهم در طراحی PUFهاست [۳۵]. همان طور که اثر انگشت هیچ دو فردی نمی‌تواند یکسان باشد، پاسخ مدل‌های مختلف PUF هم نباید به یک چالش یکسان باشد تا بتوان بر پایه آن، احراز هویت را به‌درستی انجام داد. مقدار ایده‌آل برای این معیار برابر ۵۰٪ می‌باشد و نتایج این آنالیز در جدول ۲ آمده است. با توجه به اینکه در SQ-PUF مقادیر وابسته به ترتیب هستند، برای ارزیابی این معیار در هر مدل یکسان‌بودن ترتیب در تمام حالات رعایت شده است.

٤-٥ سربار سخت‌افزاری

جدول ۳ سربار سخت‌افزاری بخش‌های مختلف یک مدل SQ-PUF را برای ۶۴ بیت چالش و پیاده‌سازی‌شده توسط FPGA نشان می‌دهد. این مدل شامل n ماژول SQ و یک RNG است و از آنجا که معمولاً RNG در سیستم‌های الکترونیکی موجود می‌باشد، می‌توان از RNG موجود در سیستم استفاده کرد و سربار سخت‌افزاری برای آن متصور نشد. بنابراین سربار سخت‌افزاری به‌صورت $n\times(\mathrm{LUT}+\mathrm{FF})$ محاسبه می‌شود؛ با توجه به اینکه برای پیاده‌سازی هر ماژول SQ به یک LUT و یک FF (فلیپ‌فلاپ) نیاز داریم. n تعداد ماژول SQ استفاده‌شده در هر PUF را نشان می‌دهد. توجه شود که LFSR و PUF را سربار سخت‌افزاری نمی‌شماریم؛ زیرا در تمام مدل‌ها به آنها نیاز داریم. برای مقایسه بهتر، نتایج یک PUF متداول[1] نیز در جدول آمده است.

نتایج، نشان‌دهنده مزیت استفاده از LFSR در مقایسه با روش متداول هستند. البته با استفاده از LFSR و با انجام تعداد مشخص از زیر چالش‌ها با یک مدار، یک پاسخ را می‌سازیم که بسیار زمان‌برتر از روش متداول است. اگرچه در روش متداول سریع‌تر به پاسخ می‌رسیم، اما نیاز به منابع سخت‌افزاری بیشتری خواهیم داشت. مدل پیشنهادی ما از منظر انرژی با توجه به تعداد کم المان‌های داخل ماژول‌های SQ (یک فلیپ فلاپ نوع D و یک XOR)، برتری زیادی به روش های متداول بحث شده دارد.

1. Basic-PUF

دارد.

۵- نتیجه‌گیری

این مقاله، یک روند جدید احراز هویت مبتنی بر PUF با نام SQ-PUF را پیشنهاد می‌کند. هدف اصلی در این کار پژوهشی، مبهم‌سازی رابطه بین جفت‌های چالش- پاسخ (CRP) برای دشوارکردن یا به‌تعویق‌انداختن فرایند مدل‌سازی یا پیش‌بینی پاسخ است که می‌تواند توسط مهاجم مورد سوءاستفاده قرار گیرد. بر این اساس با وابسته‌کردن چالش‌ها به یکدیگر، رابطه بین چالش- پاسخ‌ها را مبهم کردیم تا مهاجم نتواند وابستگی مربوط را پیش‌بینی کند. نتایج تجربی حاکی از آن است که حتی یک شبکه ANN پیچیده با مجموعه‌ای بزرگ از داده‌های چالش- پاسخ هم نمی‌تواند با دقت بیش از ۵۳٪ پاسخ‌ها را پیش‌بینی کند. با توجه به حملات مشابه به سایر PUFها می‌توان گفت مدل ارائه‌شده به‌طور قابل توجهی دارای امنیت بیشتری است. علاوه بر این، یکنواختی و یکتایی در این مدل تقریباً با مقدار ایده‌آل در Arbiter PUF یکسان باقی می‌ماند. این روش یک روند مناسب برای احراز هویت امن در دستگاه‌های IoT را فراهم می‌کند.

مراجع

[1] S. Hemavathy and V. S. Kanchana Bhaaskaran, "Arbiter PUF-a review of design, composition, and security aspects," *IEEE Access*, vol. 11, pp. 33979-34004, 2023.

[2] A. Shamsoshoara, A. Korenda, F. Afghah, and S. Zeadally, "A survey on physical unclonable function (PUF)-based security solutions for internet of things," *Computer Networks*, vol. 183, Article ID: 107593, Dec. 2020.

[3] H. Ning, F. Farha, A. Ullah, and L. Mao, "Physical unclonable function: architectures, applications and challenges for dependable security," *IET Circuits, Devices & Systems*, vol. 14, no. 4, pp. 407-424, Jul. 2020.

[4] B. Gassend, D. Lim, D. Clarke, M. Van Dijk, and S. Devadas, "Identification and authentication of integrated circuits," *Concurrency Computation Practice and Experience*, vol. 16, no. 11, pp. 1077-1098, 2004.

[5] J. W. Lee, *et al.*, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *Proc. IEEE Symp. on VLSI Circuits, Digest of Technical Papers*, pp. 176-179, Honolulu, HI, USA, 17-19 Jun. 2004.

[6] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Techniques for design and implementation of secure reconfigurable PUFs," *ACM Trans. Reconfigurable Technol Syst*, vol. 2, no. 1, Article ID: 5, 33 pp., Mar. 2009.

[7] A. Ashtari, A. Shabani, and B. Alizadeh, "A new RF-PUF based authentication of internet of things using random forest classification," in *Proc. of 16th Int. ISC Conf. on Information Security and Cryptology, ISCISC'19*, pp. 21-26, Mashhad, Iran, 28-29 Aug. 2019.

[8] B. Chatterjee, D. Das, and S. Sen, "RF-PUF: IoT security enhancement through authentication of wireless nodes using in-situ machine learning," in *Proc. of the IEEE Int. Sym. on Hardware Oriented Security and Trust, HOST'18,* pp. 205-208, May 2018.

[9] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proc. 44th ACM/IEEE Design Automation Conf.*, pp. 9-14, San Diego, CA, USA, 4-8 Jun. 2007.

[10] P. K. Sadhu and V. P. Yanambaka, "MC-PUF: a robust lightweight controlled physical unclonable function for resource constrained environments," in *Proc. of IEEE Computer Society Annual*

Symposium on VLSI, ISVLSI'22, pp. 452-453, Nicosia, Cyprus, 4-6 Jul. 2022.
[11] M. H. Ishak, M. S. Mispan, W. Y. Chiew, M. R. Kamaruddin, and M. A. Korobkov, "Secure lightweight obfuscated delay-based physical unclonable function design on FPGA," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 2, pp. 1075-1083, Apr. 2022.
[12] S. Abdolinezhad and A. Sikora, "A lightweight mutual authentication protocol based on physical unclonable functions," in *Proc. of the IEEE Int. Symp. on Hardware Oriented Security and Trust, HOST*'22, pp. 161-164, McLean, VA, USA, 27-30 2022.
[13] A. Vijayakumar and S. Kundu, "A novel modeling attack resistant PUF design based on non-linear voltage transfer characteristics," in *Proc. Design, Automation and Test in Europe, DATE'15*, pp. 653-658, Grenoble, France, 9-13 Mar. Apr. 2015.
[14] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure PUFs," in *Proc. IEEE/ACM Int. Conf. on Computer-Aided Design, Digest of Technical Papers, ICCAD'08*, pp. 670-673, San Jose, CA, USA, 10-13 Nov. 2008.
[15] D. P. Sahoo, S. Saha, D. Mukhopadhyay, R. S. Chakraborty, and H. Kapoor, "Composite PUF: a new design paradigm for physically unclonable functions on FPGA," in *Proc. of the IEEE Int. Symp. on Hardware-Oriented Security and Trust, HOST'14,* pp. 50-55, Arlington, VA, USA, 6-7 May 2014.
[16] D. E. Holcomb, W. Burleson, and K. Fu, Initial SRAM State as a Fingerprint and Source of True Random Numbers for RFID tags, 2007.
[17] P. Tuyls, *et al*., "Read-proof hardware from protective coatings," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer Verlag, pp. 369-383, Oct. 2006.
[18] M. Sauer, P. Raiola, L. Feiten, B. Becker, U. Rührmair, and I. Polian, "Sensitized path PUF: a lightweight embedded physical unclonable function," in *Proc. of the Design, Automation and Test in Europe, DATE'17,* pp. 680-685, Lausanne, Switzerland, 27-31 Mar. 2017.
[19] D. Canaday, W. A. S. Barbosa, and A. Pomerance, "A novel attack on machine-learning resistant physical unclonable functions," in *Proc. of the IEEE Int. Symp. on Hardware Oriented Security and Trust, HOST'22*, pp. 25-28, McLean, VA, USA, 27-30 Jun. 2022.
[20] J. Ye, Q. Guo, Y. Hu, H. Li, and X. Li, "Modeling attacks on strong physical unclonable functions strengthened by random number and weak PUF," in *Proc. of the IEEE VLSI Test Symposium, Computer Society*, 6 pp., San Francisco, CA, USA, 22-25 Apr. 2018.
[21] U. Rührmair and J. Sölter, "PUF modeling attacks: an introduction and overview," in *Proc. Design, Automation and Test in Europe, DATE'14,* 6 pp., Dresden, Germany, 24-28 Mar. 2014.
[22] Y. Wen and Y. Lao, "PUF modeling attack using active learning," in *Proc. IEEE Int. Symp. on Circuits and Systems*, 5 pp., Florence, Italy, 27-30 May 2018.
[23] J. Delvaux, "Security analysis of PUF-based key generation and entity authentication-KU Leuven," KU Leuven and Shanghai Jiao Tong University, 2017. Accessed: Aug. 26, 2021. [Online]. Available: https://limo.libis.be/primo-explore/fulldisplay?docid=LIRIAS1662341&context=L&vid=Lirias&search_scope=Lirias&tab=default_tab&lang=en_US&fromSitemap=1
[24] J. Delvaux, "Machine-learning attacks on PolyPUFs, OB-PUFs, RPUFs, LHS-PUFs, and PUF-FSMs," *IEEE Trans. on Information Forensics and Security*, vol. 14, no. 8, pp. 2043-2058, Aug. 2019.
[25] M. Majzoobi, M. Rostami, F. Koushanfar, D. S. Wallach, and S. Devadas, "Slender PUF protocol: a lightweight, robust, and secure authentication by substring matching," in *Proc. IEEE CS Security and Privacy Workshops, SPW'12*, pp. 33-44, San Francisco, CA, USA, 24-25 May 2012.
[26] S. T. C. Konigsmark, D. Chen, and M. D. F. Wong, "PolyPUF: physically secure self-divergence," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 7, pp. 1053-1066, Jul. 2016.
[27] J. Ye, Y. Hu, and X. Li, "RPUF: physical unclonable function with randomized challenge to resist modeling attack," in *Proc. of the IEEE Asian Hardware Oriented Security and Trust Symp.*, *Asian HOST'16*, 6 pp., Yilan, Taiwan, 19-20 Dec. 2016.
[28] Y. Gao, *et al*., "Obfuscated challenge-response: a secure lightweight authentication mechanism for PUF-based pervasive devices," in *Proc. IEEE Int. Conf. on Pervasive Computing and Communication Workshops*, *PerCom Workshops*, 6 pp., Sydney, Australia, 14-18 Mar. 2016.
[29] G. T. Becker and R. Kumar, "Active and passive side-channel attacks on delay based PUF designs," *IACR Cryptology ePrint Archive*, vol. 2014, Article ID:287, 2014, [Online]. Available: http://eprint.iacr.org/2014/287.pdf
[30] J. Shi, Y. Lu, and J. Zhang, "Approximation attacks on strong PUFs," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 2138-2151, Oct. 2020.
[31] I. G. Târşa, G. D. Budariu, and C. Grozea, "Study on a true random number generator design for FPGA," in *Proc. 8th Int. Conf. on Communications, COMM'10,* pp. 461-464, Bucharest, Romania, 10-12 Jun. 2010.
[32] T. Arciuolo and K. M. Elleithy, "Parallel, true random number generator (P-TRNG): using parallelism for fast true random number generation in hardware," in *Proc. IEEE 11th Annual Computing and Communication Workshop and Conf., CCWC'21*, pp. 987-992, NV, USA, 27-30 Jan. 2021.
[33] R. S. Durga, *et al*., "Design and synthesis of LFSR based random number generator," in *Proc. of the 3rd Int. Conf. on Smart Systems and Inventive Technology, ICSSIT*, pp. 438-442, Tirunelveli, India, 20-22 Aug. 2020.
[34] A. Maiti and P. Schaumont, "The impact of aging on a physical unclonable function," *IEEE Trans. Very Large Scale Integr VLSI Syst*, vol. 22, no. 9, pp. 1854-1864, Sept. 2014.
[35] R. L. Sembiring, R. R. Pahlevi, and P. Sukarno, "Randomness, uniqueness, and steadiness evaluation of physical unclonable functions," in *Proc. 9th Int. Conf. on Information and Communication Technology, ICoICT'2021*, pp. 429-433, Yogyakarta, Indonesia, 3-5 Aug. 2021.

سید ابوالفضل سجادی هزاوه مدرک کارشناسی مهندسی برق- الکترونیک را در سال ۱۳۹۷ از دانشگاه مهاجر اصفهان اخذ نمود و توانست در سال ۱۴۰۰ در دانشگاه تهران، مقطع کارشناسی ارشد خود را در رشته مهندسی برق- سیستم‌های الکترونیک دیجیتال را به پایان برساند و هم‌اکنون در دانشگاه لایدن هلند در حال گذراندن دوره دکتری در حوزه امنیت سخت‌افزار است. زمینه‌های تحقیقاتی مورد علاقه ایشان امنیت سخت‌افزار، شتاب‌دهنده‌های سخت‌افزاری و شبکه‌های عصبی عمیق می‌باشد.

بیژن علیزاده مدرک دکتری خود را در رشته مهندسی برق و کامپیوتر از دانشگاه تهران در سال ۱۳۸۳ دریافت کرد و از سال ۱۳۸۴ تا ۱۳۸۶ در دانشگاه صنعتی شریف به‌عنوان استادیار و از سال ۱۳۸۶ تا ۱۳۸۹ در VDEC دانشگاه توکیو به‌عنوان دانشیار پژوهشی فعالیت نمود. وی از سال ۱۳۹۰ به دانشکده مهندسی برق و کامپیوتر دانشگاه تهران پیوست و در حال حاضر، دانشیار این دانشکده است. وی بیش از ۱۳۰ مقاله در مجلات و کنفرانس‌های علمی بین‌المللی تألیف کرده و زمینه‌های تحقیقاتی او، توسعه ابزار EDA در سیستم‌های VLSI، طراحی سیستم‌های نهفته مبتنی بر FPGA، درستی‌سنجی و عیب‌یابی در سیستم‌های دیجیتال، امنیت سخت‌افزاری و سنتز سطح بالا است.