Electronical & Cyber Defence
Vol. 13, No. 1, Spring 2025, Serial No. 49

Introducing a native protocol for post-processing of experimental quantum key distribution systems in real conditions

Ali Mehri Toonabi*, Samira Momeni

Assistant Professor, Malek Ashtar University of Technology, Shahinshahr, Iran

Received:2024 /11/17, Revised: 2025/02/13, Accepted: 2025/03/13, Published: 2025/04/21

DOR: https://dor.isc.ac/dor/20.1001.1.23224347.1404.13.1.5.0

ABSTRACT

In this paper, a post-processing protocol for quantum key distribution (QKD) systems based on the widely used two-dimensional standard BB84 setup is introduced. In writing the algorithms of this protocol, the optimal version of well-known codes and functions such as LDPC encoding and decoding codes, key derivation functions, PolyHash and Toeplitz hashing functions, and one-time-pad encoding systems have been used. This protocol is not limited to use in the two-dimensional BB84 scheme, and with a simple and minor change in its sifting algorithm, it can also be used to post-process various QKD protocols (including B92, BBM92, Decoy, HD-QKD, etc.).

Keywords: Quantum cryptography, Quantum key distribution (QKD), QKD protocol, Classical post-processing, Secure key

* Corresponding Author Email: amtoonabi@gmail.com

"پدافند الکترونیکی و سایبری"
سال سیزدهم، شماره ۱،بهار ۱۴۰۴، ص۷۲-۶۱
شاپا الکترونیکی: ۸۹۷۹-۲۹۸۰ شاپا چاپی: ۴۳۴۷-۲۳۲۲

علمی – پژوهشی

معرفی یک پروتکل پساپردازش سامانه توزیع کلید کوانتومی

علی مهری تونابی*[۱]، سمیرا مومنی[۲]

۱- استادیار، ۲- پژوهشگر، دانشگاه صنعتی مالک اشتر، اصفهان، ایران

(دریافت: ۱۴۰۳/۰۸/۲۷، بازنگری: ۱۴۰۳/۱۱/۲۵، پذیرش: ۱۴۰۳/۱۲/۲۳، انتشار: ۱۴۰۴/۰۲/۰۱

DOR: https://dor.isc.ac/dor/20.1001.1.23224347.1404.13.1.5.0

چکیده

در این مقاله، یک پروتکل پساپردازش سامانه‌های توزیع کلید کوانتومی (QKD) مبتنی بر چیدمان بسیار پرکاربرد BB84 استاندارد دوبعدی معرفی شده است. در نوشتن الگوریتم‌های این پروتکل از حالت بهینه کدها و توابع شناخته شده‌ای همچون کدهای LDPC، تابع استخراج کلید، توابع درهم ساز PolyHash و Toeplitz و سیستم تولید رمز یک‌بارمصرف استفاده شده است. این پروتکل محدود به کاربرد در شمای BB84 دوبعدی نیست و با تغییر ساده و جزئی در الگوریتم غربال‌گری آن، می‌تواند برای پساپردازش پروتکل‌های QKD مختلف (از جمله B92، BBM92، Decoy، HD-QKD و...) نیز بکار رود.

کلیدواژه‌ها: رمزنگاری کوانتومی، توزیع کلید کوانتومی، پروتکل QKD، پساپردازش کلاسیک، کلید امن

۱- مقدمه

رمزنگاری کوانتومی[۱]، توسعه‌یافته‌ترین شاخه پردازش اطلاعات کوانتومی[۲] در ورود به کاربردهای مربوط به دنیای واقعی و تجربی است. رمزنگاری کوانتومی به‌عنوان یکی از اقدامات پدافند غیرعامل، به‌منظور حفاظت از اطلاعات حساس و طبقه‌بندی شده فردی، امنیتی، نظامی، صنعتی، تجاری و دولتی بسیار ضروری است. مفاهیم مقدماتی و بنیادی رمزنگاری کوانتومی در سال ۱۹۸۰ توسعه داده شد و نخستین اثبات تجربی این مفاهیم در سال ۱۹۹۲ انجام گرفت [۱].

هدف از رمزنگاری کوانتومی، فراهم‌آوردن یک شیوه قابل‌اعتماد برای انتقال کلید محرمانه و حصول اطمینان از عدم مداخله شخص سوم (اخلال‌گر) در اجرای این روش است. این روش بر مبنای قوانین بنیادی و پایه‌ای فیزیک کوانتومی پایه‌ریزی شده است. فرایند مبادله کلید محرمانه به شیوه محرمانه و سری، اصطلاحاً توزیع کلید کوانتومی[۳] (QKD) [۸-۲]نامیده می‌شود. در واقع، QKD یک فناوری امنیتی نوظهور است که از خواص و ویژگی‌های مکانیک کوانتوم جهت تبادل و اشتراک‌گذاری یک کلید محرمانه، به‌منظور ایجاد یک ارتباط با امنیت بی‌قیدوشرط استفاده می‌کند.

در رمزنگاری کوانتومی، معمولاً سه شخصیت آلیس[۴]، باب[۵] و ایو[۶] ایفای نقش می‌کنند. آلیس و باب دونفری هستند که می‌خواهند اطلاعات را بین خودشان مبادله کنند. ایو اخلال‌گری است که سعی دارد در ارسال پیام مداخله کند و بدون آن که آلیس و باب پی به حضورش ببرند اطلاعات آن‌ها را به سرقت ببرد. حال،

[1] Quantum cryptography
[2] Quantum information processing

رایانامه نویسنده مسئول: amtoonabi@gmail.com

[3] Quantum Key Distribution (QKD)
[4] Alice
[5] Bob
[6] Eve

تکلیف و مأموریت رمزنگاری کوانتومی این است کـه طرحـی ارائـه دهد که عملکرد ایو در سامانه را قابل آشکارسازی کند.

نکته قابل‌توجه آن است که رمزنگاری کوانتـومی نیـز هماننـد همتای کلاسیک خود نمی‌تواند مانع حمـلات هـک و شـنود شـود، بلکه شیوه‌ای را فراهم می‌آورد که هنگام مداخلـه شـخص سـوم در سامانه ارسال پیام، بتوان به وجود آن پی برد. این به آلـیس و بـاب اجازه می‌دهد بتوانند سیستم‌هایی برپـا کننـد کـه از طریـق آن‌هـا کلیدهای خصوصی را با اطمینـان از واقعـاً خصوصـی مانـدن آن‌هـا مبادله کنند. اگر آلیس و باب برای رمز کردن هر پیام، از یک کلیـد منحصـربه‌فـرد اسـتفاده کننـد، درواقـع توانسـته‌انـد روش رمـز یک‌بارمصرف[1] (OTP) [۹]را به طور کارا و مؤثر به کار برنـد و پیـام آن‌ها در مقابل حملات اخلال و شنود توسط ایو کاملاً ایمن خواهد بود.

برای اجرای یک پروتکل QKD، علاوه بر به‌کارگیری پدیده‌هـا و فرایندهای کوانتومی، از فرایندهای کلاسیک خاصی نیـز اسـتفاده می‌شود. لایه کوانتومی یـک پروتکـل QKD، شـامل تمـامی ادوات اپتیـک کوانتـومی و الکترواپتیکـی و نیـز تجهیـزات الکترونیکـی و مخـابراتی مـورد نیـاز بـرای راه‌انـدازی آن‌هـا اسـت. بـه مجموعـه فراینـدهای کوانتـومی بـه‌کاررفتـه در پروتکـل‌هـای QKD، انتقـال کوانتومی[2] گفته می‌شود که در آن، حالـت‌هـای کوانتـومی پـس از کدگذاری، از طریق یک کانال کوانتومی (عمدتاً فیبر نوری یا فضای آزاد) بین دو پایگاه فرستنده و گیرنده منتقـل مـی‌شـوند. ماحصـل انتقال کوانتومی، دستیابی آلیس و باب به یـک کلیـد خـام[3] اسـت. داده‌های انتقال‌یافته از طریـق لایـه کوانتـومی، بـه علـت تـأثیرات مخرب ناشی از کانال کوانتومی، نواقص و ناکاملی‌های متنوع موجود در ادوات به‌کاررفته در چیدمان سامانه و یا اخلال‌گـری توسـط ایـو دچار تغییر می‌شوند. این عامل موجب می‌شود کـه کلیـدهای خـام آلیس و باب کاملاً یکسان نباشند.

در حالـت کلـی، تغییـرات ایجـاد شـده در لایـه کوانتـومی، قابل‌شناسایی، تعیین و اصلاح هستند. این مهم با کمک مجموعه‌ای از فرایندهای کلاسیک انجام می‌گیرد که لایه پساپردازش پروتکـل QKD نامیده می‌شوند. فرایندهای کلاسـیک غربـال‌گـری[4]، اصـلاح خطا[5] (در برخی موارد حذف خطـا) و تقویـت محرمـانگی[6] (حـریم خصوصی)، اجزای جدایی‌ناپذیر هـر پروتکـل پسـاپردازش هسـتند.

فرایند اصلاح خطا مهم‌ترین بخش لایه پسـاپردازش یـک پروتکـل QKD است. یک روش متداول برای اصلاح خطا، استفاده از کدهای "بررسی توازن کـم-چگـال"[7] (LDPC) $[k,n]$ اسـت کـه در آن، n طول کدکلمه‌ها و k طول پیام‌ها برای کدگذاری هستند [۱۰و۱۱]. با اجرای درست لایه پساپردازش، علاوه بر دستیابی آلیس و باب به دو کلید کاملا یکسان، امکان ارزیابی و اعلام نظر در مورد حضور یـا عدم حضور اخلال‌گر در کانال ارتباطی نیز ممکن خواهد بـود. ایـن پروتکل باید به نحوی طراحی و پیاده‌سازی شود که اجرای آن هیچ اطلاعاتی درباره کلید نهایی فاش نکند. به عبارت دیگر، اگر ایـو بـه ارتباطات بین آلیس و باب دسترسی داشته باشـد، در ایـن صـورت نتواند هیچ اطلاعاتی از کلید محرمانه بدست آورد [۱۲].

ساختار این مقاله بدین صورت است: در بخش دوم، الگوریتم عمومی سامانه‌های QKD بیان شده است. در بخش سوم، طـرح‌واره و شمای کلی پروتکل پساپردازش پیشنهادی ارائـه شـده اسـت. در بخش چهارم، جزئیات مربوط به غربال‌گـری پروتکـل پسـاپردازش پیشنهادی بیان شده و الگوریتم‌های مورد نیـاز آن آمـده اسـت. در بخش پنجم، عملیـات مـورد نیـاز بـرای اصـلاح خطـا در پروتکـل پساپردازش پیشنهادی و جزءبه‌جزء الگوریتم‌های متفاوت آن بیـان شده است. در بخش ششـم، نحـوه تقویـت محرمـانگی در پروتکـل پساپردازش پیشنهادی ذکر شده است. نهایتاً در بخش هفتم، نتایج و دستاوردهای این ایده بیان شده است.

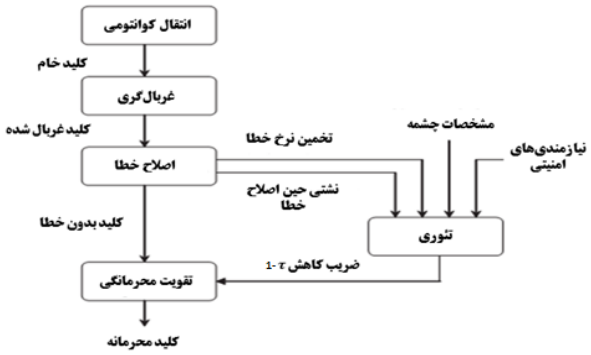
۲- الگوریتم عمومی سامانه‌های QKD

همان‌طور که بیان شد، هر پروتکـل QKD شـامل دو مرحلـه اسـت: (۱) انتقـال فوتـون‌هـا از طریـق کانـال کوانتـومی و (۲) پساپردازش از طریق کانال کلاسـیک تأییـد شـده. در مرحلـه اول، آلیس و باب یک‌رشته بیت نسبتاً مشترک اولیه به دست مـی‌آورنـد که در مورد امنیت آن نمی‌توان اظهـارنظر قطعـی کـرد. در مرحلـه دوم، با انجام غربال گری، تخمین پـارامتر، تصـحیح خطـا و تقویـت حریم خصوصی در یک کانال کلاسیک تأیید شده، آلیس و باب یک کلید یکسان و با امنیت بدون قیدوشرط را از کلید اولیـه اسـتخراج می‌کنند [۱۳].

گام‌های اصـلی و ضـروری در اجـرای هـر الگـوریتم QKD در شکل (۱) بیان شده است.

[7] One-Time-Pad (OTP)
[1] Quantum transmission
[3] Raw key
[3] Sifting
[4] Error correction
[5] Privacy amplification

[7] Low-Density Parity Check (LDPC)



شکل (۱). طرحواره الگوریتم عمومی سامانه‌های QKD [۱۴]

به طور خلاصه، برای اجرای یک پروتکل QKD مبتنی بر آماده‌سازی-اندازه‌گیری[1] (P&M) [۱۵]، ابتدا آلیس با توجه به رشته بیت تصادفی و پایه‌های آماده‌سازی تصادفی که در اختیار می‌گیرد، دنباله‌ای از پالس‌های تک‌فوتونی را که در حالت‌های کوانتومی مناسب آماده شده‌اند، به ترتیب از طریق یک کانال کوانتومی مناسب برای باب ارسال می‌کند. باب فوتون‌ها را دریافت کرده، با انتخاب تصادفی پایه‌های اندازه‌گیری، حالت کوانتومی هر پالس را استخراج و کیوبیت متناظر با آن حالت را ثبت می‌کند. پس از انتقال کوانتومی تمام رشته بیت اولیه، با استفاده از یک کانال عمومی (کلاسیک)، پایه‌های متناظر انتخاب شده توسط آلیس و باب برای هر پالس با هم مقایسه و موقعیت پالس‌هایی که در آن‌ها پایه‌های مشابه انتخاب شده، به عنوان تنها پالس‌هایی که باید حفظ شوند، مشخص و سایر پالس‌ها کنار گذاشته می‌شوند. در یک سامانه‌ی ایده‌آل، اگر ایو حاضر نباشد، می‌توان از مجموعه بیت به دست آمده به عنوان یک کلید امن استفاده کرد؛ اما طبیعتاً آلیس و باب نمی‌توانند از همان ابتدا بنا را بر عدم حضور ایو بگذارند، بلکه می‌بایست با استفاده از رشته بیت حاصل شده و انجام یک ارتباط کلاسیک، امکان حضور ایو را بررسی کنند. این کار با اعلان عمومی و مقایسه تعدادی از بیت‌های متناظر در کلیدهای آلیس و باب و محاسبه‌ی نرخ خطای موجود در آن‌ها انجام می‌گیرد. واضح است که بیت‌های فاش شده می‌بایست به کلی کنار گذاشته شده و وارد کلید نهایی نشوند. در اصل، وجود حتی یک خطا، حضور ایو در سامانه را آشکار و کلید به دست آمده را ناامن و غیرقابل استفاده می‌کند؛ اما تمام سامانه‌های ارتباطی واقعی ذاتاً دارای مقداری خطا هستند. بنابراین وجود تعدادی خطا در رشته بیت ساخته شده با استفاده از کانال کوانتومی، حتی در صورت عدم مداخله ایو، غیر قابل اجتناب است. با این حال، از آن‌جا که فرض می‌شود ایو بتواند سامانه دارای خطای ذاتی e آلیس و باب را با یک سامانه بدون خطا تعویض کند، وجود هر گونه خطا در کلید غربال شده به فعالیت ایو نسبت داده می‌شود. این خطای اندازه‌گیری شده را اصطلاحاً نرخ خطای باب[2] (خطای ناشی از مداخله ایو در نتایج اندازه‌گیری باب)، e_B، می‌نامند [۱۴]. اگر نرخ خطای کلید غربال شده، از یک مقدار آستانه (بیشینه نرخ خطای قابل تحمل توسط سامانه، $e_{B_{max}}$، که قبل از اجرای پروتکل، به صورت نظری و با توجه به نوع پروتکل و ملاحظه نوع خاصی از حمله توسط ایو مشخص می‌شود) کم‌تر باشد، آلیس و باب به بهای فاش شدن تعداد دیگری از بیت‌های موجود، قادر به اصلاح (اصلاح یا حذف) این خطاها خواهند بود. در صورت اصلاح خطاها، طول کلیدهای غربال شده و اصلاح شده با هم برابر است، در حالی‌که با حذف خطاها، طبیعتاً طول کلید اصلاح شده کم‌تر از کلید غربال شده خواهد بود. در پایان فرایند اصلاح خطا، آلیس و باب به یک رشته بیت مشترک دست پیدا می‌کنند که ممکن است طول آن به میزان قابل توجهی کوتاه‌تر از کلید اولیه باشد. کار باقی‌مانده، تخمین و ترجیحاً قرار دادن یک حد بالا بر روی میزان اطلاعاتی است که یک اخلال‌گر احتمالی می‌تواند از بیت‌های باقی‌مانده داشته باشد. فرایند تقویت محرمانگی به آلیس و باب این امکان را می‌دهد که با فشرده کردن طول کلید و کاهش دادن بیشینه احتمال شناسایی درست هر بیت توسط ایو، اطلاعات ایو از کلید نهایی را به حدی که کاملاً قابل چشم‌پوشی است برسانند. با تعریف ضریب کاهش[3]، $1-\tau$، به صورت کسر بیت اطلاعات نشت کرده به ایو به ازای یک بیت مبادله شده بین آلیس و باب [۱۴]، میزان فشردگی مورد نیاز در فرایند تقویت محرمانگی با توجه به τ تعیین می‌شود، که خود، تابع نرخ خطای ذاتی سامانه و خطای ناشی از حمله ایو است.

۳- شمای کلی پروتکل پساپردازش پیشنهادی

پس از اجرای گام انتقال کوانتومی، آلیس کلید خام K_{raw}^A و پایه‌های آماده‌سازی b^A را در اختیار دارد. K_{raw}^A و b^A دو رشته

[1] Prepare-Measure (P&M)

[2] Bob's error rate
[2] Shrinking factor

دوتایی[1] با طول یکسان هستند. باب نیز کلید خام K_{raw}^{B} و پایه‌های اندازه‌گیری b^B را در اختیار دارد. برخلاف K_{raw}^{A}، K_{raw}^{B} یک رشته سه تایی[2] از اعداد $\{-1,0,1\}$ است، اما b^B به صورت یک رشته دوتایی است و طول آن با طول K_{raw}^{B} برابر می‌باشد. توجه شود که در کلید خام K_{raw}^{B}، اگر عضو t-ام برابر -1 باشد، بدین معناست که در زمان t باب هیچ پالسی از آلیس دریافت نکرده است.

برای اجرای پروتکل پساپردازش پیشنهادی، سه فرایند غربال‌گری، اصلاح خطا و تقویت محرمانگی به ترتیب و به‌صورت زیر انجام می‌شوند:

(۱) فرایند غربال‌گری

(۲) فرایند اصلاح خطا، شامل:

(۲-۱) مرحله کدگذاری

(۲-۲) مرحله کدگشایی

(۲-۳) مرحله صحت‌سنجی کدگشایی

(۲-۴) مرحله تخمین پارامتر

(۳) فرایند تقویت محرمانگی.

پس از اجرای موفقیت‌آمیز فرایندهای فوق، آلیس و باب در نهایت به کلید محرمانه K_{sec} دست پیدا می‌کنند.

۴- فرایند غربال‌گری در پروتکل پساپردازش پیشنهادی

در مرحله اول پروتکل پساپردازش، آلیس و باب الگوریتم غربال‌گری را اجرا می‌کنند. با اجرای این مرحله، آلیس به کلید K_{sift}^{A} و باب به کلید K_{sift}^{B} دست پیدا می‌کند. کلیدهای غربال شده K_{sift}^{A} و K_{sift}^{B} به صورت رشته‌بیت‌هایی با طول یکسان هستند ولی با توجه به منابع خطای معرفی شده در بخش ۱، برخی از بیت‌های آن‌ها با هم متفاوت است.

در فرایند غربال‌گری، آلیس ابتدا الگوریتم SiftA1 را مطابق جدول (۱) اجرا می‌کند. هدف از اجرای این الگوریتم، تولید رشته بیت $Stag^A$ جهت احراز اصالت b^A است. رشته بیت $Stag^A$ در اینجا تگ نامیده می‌شود. ورودی الگوریتم SiftA1 رشته بیت b^A و کلید K است. کلید K کلید اصلی است که پیش از اجرای پروتکل بین آلیس و باب به اشتراک گذاشته می‌شود. با استفاده از کلید K و تابع استخراج کلید[3] (KDF)، زیرکلیدهای $subk_i$ تولید می‌شوند. سپس تابع درهم‌ساز PolyHash [۱۶] که در ادامه به اختصار PH نامیده می‌شود، بر روی $(subk_i, b_i^A)$ اثر کرده، رشته بیت tag_i^A تولید می‌شود. در نهایت، تگ آلیس به صورت $Stag^A = tag_1^A \| \ldots \| tag_l^A$ تولید می‌شود.

در این الگوریتم، $len\ (b_i^A)$ بیان کننده طول (تعداد بیت‌ها) هر یک از بلوک‌ها، پس از تقسیم رشته بیت b^A به l بلوک است. به طور کلی منظور از $len(x)$، طول (تعداد بیت‌های) رشته بیت x است. برای این‌که احتمال تصادم (تولید دو رشته بیت با تگ‌های یکسان) بسیار پایین باشد، شرط $len\ (b_i^A) \leq 2^{14} - 1$ قرار داده شده است [۱۶]. بعد از اجرای الگوریتم SiftA1، آلیس b^A و $Stag^A$ را برای باب ارسال می‌کند.

جدول (۱). گام اول غربال‌گری آلیس (الگوریتم SiftA1). در این الگوریتم، پایه‌های آماده‌سازی آلیس برای باب ارسال می‌شود.

ورودی الگوریتم: b^A و کلید اصلی K **خروجی الگوریتم: رشته بیت $Stag^A$**
۱- رشته بیت b^A را به l بلوک با طول دلخواه و با شرط $len\ (b_i^A) \leq 2^{14} - 1$ تقسیم کن. در این صورت، رشته بیت b^A از کنار هم قرار گرفتن (الحاق) l بلوک ساخته می‌شود $(b_1^A \| \ldots \| b_l^A \leftarrow b^A)$،
۲- $Counter \leftarrow 1$،
۳- برای $i \in \{1, \ldots, l\}$ مراحل زیر را انجام بده:
۳-۱- $subk_i \leftarrow \mathrm{KDF}(K, Counter)$،
۳-۲- $tag_i^A \leftarrow \mathrm{PH}(subk_i, b_i^A)$ ،
۳-۳- $Counter \leftarrow Counter + 1$،
۴- $Stag^A = tag_1^A \| \ldots \| tag_l^A$،
۵- $Stag^A$ را به عنوان خروجی تولید کن.

باب بعد از دریافت b^A و $Stag^A$، الگوریتم SiftB را مطابق جدول (۲) اجرا می‌کند.

جدول (۲). غربال‌گری باب (الگوریتم SiftB). در این الگوریتم، کلید غربال شده باب تولید می‌شود.

ورودی الگوریتم: b^A ، $Stag^A$، b^B، K_{raw}^B و کلید اصلی K **خروجی الگوریتم: بیت $v = 0$ یا رشته بیت‌های $Stag^B, res$ و K_{sift}^B**
۱- رشته بیت b^A را به l بلوک با طول len تقسیم کن $(b_1^A \| \ldots \| b_l^A \leftarrow b^A)$،
۲- $Counter \leftarrow 1$،
۳- $c \leftarrow 0$ و $tem \leftarrow [\,]$،

[1] Binary
[2] Ternary
[3] Key Derivation Function (KDF)

۴- برای $i \in \{1,2,\dots,l\}$ مراحل زیر را انجام بده:
۴-۱- $subk_i \leftarrow \mathrm{KDF}(K, Counter)$،
۴-۲- $tag'_i \leftarrow \mathrm{PH}(subk_i, b_i^A)$ ،
۴-۳- $Counter \leftarrow Counter + 1$ ،
۴-۴- اگر $tag'_i \neq tag_i^A$، در این صورت $c \leftarrow c + 1$،
۵- اگر $c \geq 1$، در این صورت $v \leftarrow 0$ را به عنوان خروجی تولید کن،
در غیر این صورت مراحل زیر را انجام بده:
۶- $Len \leftarrow len(K_{raw}^B)$ و $res \leftarrow \{0\}^{Len}$،
۷- برای $i \in \{1,2,\dots,Len\}$ مراحل زیر را انجام بده:
۷-۱- اگر $K_{raw,i}^B \neq -1$ و $b_i^A = b_i^B$، در این صورت:
۷-۲- $res_i \leftarrow 1$،
۷-۳- $tem \leftarrow tem \| K_{raw,i}^B$،
۸- برای $i \in \{1,\dots,l\}$ مراحل زیر را انجام بده:
۸-۱- $subk_i \leftarrow \mathrm{KDF}(K, Counter)$،
۸-۲- $tag_i^B \leftarrow \mathrm{PH}(subk_i, res_i)$ که $res = res_1 \| \dots \| res_l$ است،
۸-۳- $Counter \leftarrow Counter + 1$،
۹- $Stag^B = tag_1^B \| \dots \| tag_l^B$،
۱۰- $K_{sift}^B \leftarrow tem$.
۱۱- رشته بیت‌های $Stag^B$، res و K_{sift}^B را به عنوان خروجی تولید کن.

با اجرای الگوریتم SiftB، باب ابتدا پایه‌های دریافت شده از آلیس (یعنی رشته بیت b^A) را با کمک تابع درهم‌ساز PH احراز اصالت می‌کند. اگر یکی از بلوک‌های b^A توسط ایو تغییر داده شده باشند، در این صورت متغیر $c \geq 1$ خواهد بود و در نتیجه الگوریتم $v = 0$ را به عنوان خروجی تولید خواهد کرد. در صورتی که $c = 0$ باشد، اجرای الگوریتم ادامه پیدا خواهد کرد. در ادامه الگوریتم، اگر $K_{raw,i}^B \neq -1$ (یعنی اگر i–امین پالس ارسالی آلیس توسط باب دریافت شده باشد) و $b_i^A = b_i^B$ (یعنی برای i–امین پالس، پایه‌های آلیس و باب یکسان باشند) باشد، الگوریتم بیت شماره i از رشته بیت K_{raw}^B را به عنوان یکی از بیت‌های کلید K_{sift}^B انتخاب خواهد کرد و درایه i–ام بردار res را برابر با 1 قرار خواهد داد. اگر درایه i–ام بردار res برابر با 1 باشد، نشان می‌دهد که باب بیت i–ام رشته K_{raw}^B را انتخاب کرده و آن را یکی از بیت‌های کلید K_{sift}^B قرار داده است. در ادامه الگوریتم SiftB، تابع درهم‌ساز PH بر روی res اعمال و تگ $Stag^B$ تولید می‌شود. بعد از اجرای الگوریتم SiftB، اگر خروجی الگوریتم به صورت $v = 0$ باشد، باب اجرای پروتکل را قطع می‌کند، در غیر این صورت، باب رشته بیت‌های $Stag^B$ و res را برای آلیس ارسال می‌کند.

آلیس بعد از دریافت رشته بیت‌های $Stag^B$ و res، الگوریتم SiftA2 را مطابق جدول (۳) اجرا می‌کند. برای این کار، آلیس ابتدا رشته‌بیت res را احراز اصالت می‌کند. اگر ایو برخی بلوک‌های رشته‌بیت res را تغییر داده باشد، در این صورت $c \geq 1$ خواهد بود و در نتیجه الگوریتم خروجی $v = 0$ را تولید و آلیس اجرای پروتکل را قطع خواهد کرد. اما اگر $c = 0$ باشد، در این صورت الگوریتم با استفاده از رشته بیت‌های res و K_{raw}^A، کلید غربال‌شده K_{sift}^A را محاسبه و تولید خواهد کرد. مراحل مختلف اجرای الگوریتم SiftA2 به صورت زیر است. توجه شود که در اینجا مقدار ابتدایی $Counter$ از خروجی الگوریتم SiftA1 تعیین می‌شود که برابر با $l + 1$ است.

جدول (۳). گام دوم غربال‌گری آلیس (الگوریتم SiftA2). در این الگوریتم، کلید غربال شده آلیس تولید می‌شود.

ورودی الگوریتم: res، $Stag^B$، K_{raw}^A، کلیداصلی K و شمارنده $Counter$ **خروجی الگوریتم: بیت $v \in \{0,1\}$ یا رشته بیت K_{sift}^A**
۱- $c \leftarrow 0$، $res_1 \| \dots \| res_l \leftarrow res$ و $tag_1^B \| \dots \| tag_l^B \leftarrow Stag^B$، $tem \leftarrow [\,]$.
۲- $Len \leftarrow len(K_{raw}^A)$.
۳- برای $i \in \{1,2,\dots,l\}$ مراحل زیر را انجام بده:
۳-۱- $subk_i \leftarrow \mathrm{KDF}(K, Counter)$،
۳-۲- $tag_i'^A \leftarrow \mathrm{PH}(subk_i, res_i)$ ،
۳-۳- $Counter \leftarrow Counter + 1$ ،
۳-۴- اگر $tag_i^B \neq tag_i'^A$، در این صورت $c \leftarrow c + 1$،
۴- اگر $c \geq 1$، در این صورت، $v \leftarrow 0$ را به عنوان خروجی تولید کن، در غیر این صورت، مراحل زیر را انجام بده:
۵- برای هر $i \in \{1,2,\dots,Len\}$ اگر $res_i = 1$، در این صورت، $tem \leftarrow tem \| K_{raw,i}^A$،
۶- $K_{sift}^A \leftarrow tem$ را به عنوان خروجی تولید کن.

بعد از اجرای فرایند غربال‌گری، فرایند اصلاح خطای پروتکل پساپردازش انجام می‌شود. جزئیات انجام این فرایند، در بخش بعد آمده است.

۵- فرایند اصلاح خطا در پروتکل پساپردازش پیشنهادی

همان‌طور که در بخش قبل گفته شد، فرایند اصلاح خطای پروتکل پساپردازش پیشنهاد شده در این مقاله، شامل چهار مرحله مختلف است که به ترتیب عبارت‌اند از:

(۱) کدگذاری
(۲) کدگشایی
(۳) صحت‌سنجی کدگشایی
(۴) تخمین پارامتر.

جزئیات مربوط به اجرای هر یک از چهار مرحله مربوط به فرایند اصلاح خطا و الگوریتم‌های هر یک از آن‌ها در ادامه بیان شده است.

۵-۱- کدگذاری

در این مرحله، کلید K_{sift}^B توسط باب مطابق جدول (۴) کدگذاری می‌شود.

جدول (۴). گام اول کدگذاری (الگوریتم Encoding). در این الگوریتم، کلید غربال شده باب کدگذاری می‌شود.

ورودی الگوریتم: K_{sift}^B، کلیداصلی K و شمارنده $Counter$ **خروجی الگوریتم: پیام کدگذاری شده باب m^B**
۱- کلید K_{sift}^B را به l بلوک تقسیم کن: $(K_{sift,1}^B \| \ldots \| K_{sift,l}^B \leftarrow K_{sift}^B)$،
۲- برای هر $i \in \{1, \ldots, l\}$ مراحل زیر را انجام بده:
۲-۱- $subk_i \leftarrow \mathrm{KDF}(K, Counter)$،
۲-۲- $m_i \leftarrow LDPCEncode(K_{sift,i}^B)$،
۲-۳- $m_i^B \leftarrow OTPE(m_i, subk_i)$،
۲-۴- $Counter \leftarrow Counter + 1$،
۳- $m^B = m_1^B \| \ldots \| m_l^B$ را به عنوان خروجی تولید کن.

در اینجا برای کدگذاری، از الگوریتم کدگذاری شناخته شده LDPC $-[k,n]$ استفاده شده است. از آنجا که در این مرحله، تابع PH بر روی بلوک‌های K_{sift}^B اثر می‌کند و از آن برای احراز اصالت استفاده می‌شود، بنابراین باید مقادیر k و n به نحوی انتخاب شوند که با پارامترهای تابع PH مطابقت داشته باشند. به عبارت دیگر، باید طول بلوک‌های LDPC به نحوی انتخاب شوند که بتوان تابع PH را بر روی آن‌ها اثر داد. توجه شود که اگر شرط $Len \leq 2^{14} - 1$ برقرار باشد، این مطابقت وجود خواهد داشت.

مطابق الگوریتم Encoding (جدول ۴)، بعد از کدگذاری بلوک‌های کلید غربال شده K_{sift}^B، هر کدام از کدکلمه‌ها با استفاده از سیستم رمز یک‌بارمصرف $OneTimePadEncrypt$ (که در ادامه، با $OTPE$ نمایش داده می‌شود) رمز می‌شوند. پس از آن، بلوک‌های حاصل از پیام‌های رمزشده با هم الحاق می‌شوند و پیام کدگذاری شده باب m^B را تشکیل می‌دهند. برای رمزگذاری از زیرکلید $subk = subk_1 \| \ldots \| subk_l$ استفاده می‌شود که طول آن برابر با n است. زیرکلید $subk$ با استفاده از کلید اصلی K و تابع KDF به صورت $subk = KDF(K, Counter)$ تولید می‌شود که در آن، شمارنده Counter باعث می‌شود که خروجی‌های تکراری توسط KDF تولید نشوند.

فرآیند رمزگذاری توسط سیستم رمز $OTPE$ به‌صورت زیر است:

$$OTPE(m_i, subk_i) = m_i \oplus subk_i. \quad (۱)$$

فرآیند رمزگشایی نیز که با نماد $OneTimePadDecrypt(.)$ (با علامت اختصاری $OTPD$) نشان داده می‌شود، مشابه فرآیند رمزگذاری خواهد بود.

در گام بعدی از مرحله کدگذاری، باب تابع درهم از PH را مطابق جدول (۵)، بر روی هر کدام از بلوک‌های $K_{sift}^B = K_{sift,1}^B \| \ldots \| K_{sift,l}^B$ اعمال می‌کند و سپس تگ $au^B = au_1^B \| \ldots \| au_l^B$ را برای احراز اصالت تولید می‌کند.

نحوه اعمال تابع درهمساز PH بر روی بلوک $K_{sift,j}^B$ به صورت زیر است:

$$\begin{aligned} au_j = PH\left(K_{sift,j}^B, subk_j\right) \\ = \left(a_{l,j} + a_{l-1,j} subk_j + \cdots \right. \\ \left. + subk_j^{l-1} a_{1,j}\right) mod(p). \end{aligned} \quad (۲)$$

که در آن، $a_{1,j} \| a_{2,j} \| \ldots \| a_{l,j} \leftarrow K_{sift,j}^B$ و $p = 2^{32} - 5$. $subk_j$ نیز یک زیرکلید ۳۲ بیتی است که با استفاده از کلید اصلی K و به صورت $subk_j = KDF(K, Counter)$ تولید می‌شود.

جدول (۵). گام دوم کدگذاری (الگوریتم PH). در این الگوریتم، تگ احراز اصالت برای پیام حاوی کلید غربال شده کدگذاری شده باب تولید می‌شود.

ورودی الگوریتم: K_{sift}^B، کلید اصلی K و شمارنده $Counter$ **خروجی الگوریتم: تگ au^B**
۱- رشته بیت K_{sift}^B را به l بلوک با طول یکسان تقسیم کن: $(K_{sift,1}^B \| \ldots \| K_{sift,l}^B \leftarrow K_{sift}^B)$،
۲- برای هر $i \in \{1, \ldots, l\}$ مراحل زیر را انجام بده:
۲-۱- $a_1 \| \ldots \| a_{32} \leftarrow K_{sift,i}^B$،
۲-۲- $y \leftarrow 0$،
۲-۳- $subk_i \leftarrow \mathrm{KDF}(K, Counter)$،
۲-۴- $k' \leftarrow subk_i$،
۲-۵- $y \leftarrow k'y + a_j \bmod (p)$ برای $j = 1,2, \ldots, 32$،
۲-۶- $au_i^B \leftarrow y$،
۲-۷- $Counter \leftarrow Counter + 1$،
۳- $au^B = au_1^B \| \ldots \| au_l^B$ را به عنوان خروجی تولید کن.

توجه شود که چون عدد p در الگوریتم PH یک عدد ۳۲ بیتی است، به همین دلیل $K_{sift,i}^B$ به صورت الحاق ۳۲ رشته بیت کوچکتر نوشته شده است. اگر در الگوریتم PH عدد p یک عدد اول ۶۴ بیتی انتخاب شود، در این صورت $K_{sift,i}^B$ باید به صورت الحاق ۶۴ رشته بیت کوچکتر نوشته شود. این کار احتمال حمله تصادم و جعل را کاهش خواهد داد ولی سرعت پیاده‌سازی الگوریتم را کمی کندتر می‌کند. در اینجا نیز بعد از هر بار اجرای تابع KDF، شمارنده

Counter به صورت $Counter = Counter + 1$ بروزرسانی می‌شود تا در هر اجرا، یک زیرکلید جدید تولید شود. همچنین برای هر بلوک $K^B_{sift,j}$ از یک زیرکلید متمایز استفاده می‌شود.

تابع درهمساز PH در برابر حملات تصادم و جعل امنیت بالایی را فراهم می‌کند و اثبات شده است که اگر $z = (z_1, \dots, z_l)$ و $z' = (z'_1, .., z'_l)$ دو رشته بیت متفاوت باشند، در این صورت رابطه زیر به ازای آن‌ها برقرار است [۱۶]:

$$\{\Pr[\mathrm{PH}(z, subk_j) = \mathrm{PH}(z', subk_j), subk_j \in \{0,1,\dots,p-1\}]\} \le \frac{l-1}{p} \quad (۳)$$

بنابراین، می‌توان گفت که اگر در اجرای تابع درهمساز PH از زیر کلیدهای متمایز استفاده شود، در این صورت این تابع درهمساز امنیت بالایی را در برابر حمله تصادم و جعل فراهم می‌کند.

همان‌طور که گفته شد، استفاده از رابطه (۲) برای محاسبه PH، باعث کاهش سرعت پیاده‌سازی پروتکل خواهد شد. برای رفع این مشکل، می‌توان رابطه (۲) را به صورت زیر نوشت:

$$au_j = a_{l,j} + subk_j\Big(a_{l-1,j} + subk_j\big(a_{l-2,j} + subk_j\big(a_{l-3,j} + \cdots\big)\big)\Big). \quad (۴)$$

بنابراین، برای محاسبه $\mathrm{PH}(K^B_{sift,j}, subk_j)$ کافیست عملیات زیر را $l-1$ مرتبه انجام داد:

$$y \leftarrow 0, \qquad y \leftarrow subk_j\, y + a_i \bmod(p). \quad (۵)$$

اکنون با استفاده از روابط بالا، می‌توان الگوریتم PH را به‌صورت جدول ۵ نوشت:

باب بعد از تولید m^B و au^B، آن‌ها را برای آلیس ارسال می‌کند.

۵-۲- کدگشایی

آلیس بعد از دریافت پیام (m^B, au^B)، ابتدا با استفاده از یک فرآیند کدگشایی، پیام m^B را کدگشایی می‌کند. این فرایند در جدول (۶) با نام الگوریتم Decoding آورده شده است. مطابق جدول ۶، برای اجرای الگوریتم Decoding ابتدا زیربلوک‌های رشته بیت m^B توسط سیستم رمز یک‌بار مصرف $OTPE$ رمزگشایی می‌شوند. سپس بلوک‌های رمزگشایی شده، با استفاده از $LDPCDecode$ کدگشایی می‌شوند و رشته بیت m^A به عنوان خروجی الگوریتم تولید می‌شود.

جدول (۶). گام اول کدگشایی (الگوریتم Decoding). در این الگوریتم، پیام کدگذاری شده باب توسط آلیس کدگشایی می‌شود.

ورودی الگوریتم: m^B، کلید اصلی K و شمارنده $Counter$ **خروجی الگوریتم: پیام کدگشایی شده آلیس m^A**
۱- پیام m^B را به l بلوک با طول یکسان تقسیم کن: $(m^B_1 \| \dots \| m^B_l \leftarrow m^B)$،
۲- برای $i \in \{1, \dots, l\}$ مراحل زیر را انجام بده:
۲-۱- $subk_i \leftarrow \mathrm{KDF}(K, Counter)$،
۲-۲- $b_i \leftarrow OTPD_{(m^B_i, subk_i)}$،
۲-۳- $m^A_i \leftarrow \mathrm{LDPCDecode}(b_i)$،
۲-۴- $Counter \leftarrow Counter + 1$،
۳- $m^A = m^A_1 \| \dots \| m^A_l$ را به عنوان خروجی تولید کن.

آلیس بعد از اجرای الگوریتم Decoding، الگوریتم Corr را مطابق جدول (۷) اجرا می‌کند. این الگوریتم رشته بیت $m^A = m^A_1 \| \dots \| m^A_l$ و $K^A_{sift} = K^A_{sift,1} \| \dots \| K^A_{sift,l}$ را به عنوان ورودی می‌گیرد و با مقایسه بلوک‌های آن‌ها رشته بیت $corr = corr_1 \| \dots \| corr_l$ را تولید می‌کند. برای هر $i \in \{1, \dots, l\}$ اگر $K^A_{sift,i} = m^A_i$ باشد، در این صورت الگوریتم $corr_i = K^A_{sift,i}$ را قرار می‌دهد، در غیر این صورت $corr_i$ یک رشته بیت تمام یک قرار داده می‌شود.

جدول (۷). گام دوم کدگشایی (الگوریتم Corr). در این الگوریتم، پیام کدگشایی شده و کلید غربال شده آلیس با هم مقایسه می‌شود.

ورودی الگوریتم: رشته بیت‌های m^A و K^A_{sift} **خروجی الگوریتم: رشته بیت $corr$**
۱- رشته بیت‌های m^A و K^A_{sift} را به l بلوک تقسیم کن: $(K^A_{sift,1} \| \dots \| K^A_{sift,l} \leftarrow K^A_{sift}$ و $m^A_1 \| \dots \| m^A_l \leftarrow m^A)$،
۲- برای هر $i \in \{1, \dots, l\}$ اگر $K^A_{sift,i} = m^A_i$ در این صورت $corr_i \leftarrow K^A_{sift,i}$،
۳- برای هر $i \in \{1, \dots, l\}$ اگر $K^A_{sift,i} \neq m^A_i$ در این صورت $corr_i \leftarrow \overbrace{11\dots11}^{k}$،
۴- $corr = corr_1 \| \dots \| corr_l$ را به عنوان خروجی تولید کن.

بعد از اجرای الگوریتم Corr، آلیس الگوریتم $VerifiedBlocks1$ را مطابق جدول (۸) و به ازای ورودی $(corr, au^B)$ اجرا می‌کند.

جدول (۸). گام سوم کدگشایی (الگوریتم VerifiedBlocks1). در این الگوریتم، کلید تایید شده آلیس تولید می‌شود.

ورودی الگوریتم: $corr$، رشته بیت au^B، کلید اصلی K و شمارنده $Counter$

خروجی الگوریتم: کلید تأیید شده K_{Ver} و مجموعه Ver که شامل اندیس‌های متعلق به بلوک‌های تأیید شده است

۱- $Ver \leftarrow [\,]$ و $K_{Ver} \leftarrow [\,]$،

۲- رشته بیت‌های $corr$ و au^B را به l بلوک تقسیم کن:

($corr_1 \| \ldots \| corr_l \leftarrow corr$ و $au_1^B \| \ldots \| au_l^B \leftarrow au^B$)،

۳- برای هر $i \in \{1, \ldots, l\}$ مراحل زیر را انجام بده:

۳-۱- $subk_i \leftarrow \mathrm{KDF}(K, Counter)$،

۳-۲- $au_i^A \leftarrow \mathrm{PH}(corr_i, subk_i)$،

۳-۳- اگر $au_i^A \neq au_i^B$ در این صورت $K_{Ver} = K_{Ver} \| corr_i$ و $Ver = Ver \cup \{i\}$،

۳-۴- $Counter \leftarrow Counter + 1$،

۴- K_{Ver} و Ver را به عنوان خروجی تولید کن.

این الگوریتم تابع درهمساز PH را بر روی هر کدام از بلوک‌های $corr = corr_1 \| \ldots \| corr_l$ اعمال می‌کند. در صورتی که $\mathrm{PH}(subk_i, corr_i) = au_i^B$ باشد، باب نتیجه می‌گیرد که بلوک $corr_i$ به‌درستی کدگشایی شده و خطای احتمالی موجود در بیت‌های آن به درستی اصلاح شده است. به این نوع از بلوک‌ها، بلوک‌های تأیید شده گفته می‌شود. در حقیقت $corr_i$ بلوکی است که آلیس و باب هر دو آن را در اختیار خواهند داشت. آلیس با استفاده از بلوک‌های تأیید شده، به رشته بیت K_{Ver} دست پیدا می‌کند که به آن کلید تأیید شده گفته می‌شود.

آلیس بعد از محاسبه کلید تأیید شده K_{Ver} و مجموعه Ver تابع درهمساز توپلیتز[1] (Toeplitz) را بر روی Ver اعمال می‌کند و تگ $Ttag$ را جهت احراز اصالت بدست می‌آورد. تابع درهمساز توپلیتز به صورت زیر عمل می‌کند:

$$Ttag = h(m_{au}) = T_s\, m_{au} \oplus r, \qquad (۶)$$

که در آن، m_{au} نمایش برداری پیام مورد نظر برای احراز اصالت است که در این جا برابر با مجموعه Ver خواهد بود (دقت شود که درایه‌های بردار m_{au} به صورت دوتایی هستند). $T_s = [T_{i,j}]$ یک ماتریس دوتایی توپلیتز از اندازه $l_h \times l_{m_{au}}$ است که در آن $l_{m_{au}}$ برابر با طول بردار m_{au} است. در نهایت، r یک بردار دوتایی محرمانه از طول l_h است.

ماتریس T_s از اندازه $l_h \times l_{m_{au}}$ را یک ماتریس دوتایی توپلیتز گویند اگر درایه‌های آن در رابطه زیر صدق کنند:

$$T_{i,j} = T_{i+1,j+1} \in \{0,1\} \qquad \begin{matrix} i = 1,2, \ldots, l_h\,, \\ j = 1,2, \ldots, l_{m_{au}}. \end{matrix} \qquad (۷)$$

به‌عنوان‌مثال، ماتریس زیر یک ماتریس توپلیتز از اندازه 4×8 است:

$$\begin{pmatrix} 01101000 \\ 00110100 \\ 10011010 \\ 11001101 \end{pmatrix}.$$

برای تولید ماتریس T_s، آلیس به بردار دوتایی $S = (s_{1-l_h}, s_{1-l_h+1}, \ldots, s_{l_{m_{au}}-1})$ نیاز دارد و با استفاده از رابطه زیر می‌تواند ماتریس T_s را به راحتی تولید کند:

$$T_{i,j} = T_{i+1,j+1} = s_{j-i}, \quad \text{for} \quad \begin{matrix} i = 1, \ldots, l_h, \\ j = 1, \ldots, l_{m_{au}}. \end{matrix} \qquad (۸)$$

تابع درهمساز توپلیتز امنیت مناسبی را برای پروتکل پساپردازش فراهم می‌کند و اثبات شده است که در صورت استفاده از آن، احتمال موفقیت‌آمیز بودن حملات تصادم و جعل حداکثر برابر با 2^{-l_h} است و $l_h \geq 64$ یک انتخاب مناسب خواهد بود [۱۷]. برای استفاده از تابع درهمساز توپلیتز، می‌بایست آلیس و باب پیش از اجرای پروتکل پساپردازش مقادیر l_h و $l_{m_{au}}$ را مشخص کنند. به این مقادیر پارامترهای تابع درهمساز توپلیتز گفته می‌شود. آلیس و باب می‌توانند بردارهای محرمانه S و r را با استفاده از کلید اصلی K و تابع تولید کلید KDF به صورت زیر تولید کنند:

$$r \| S = \mathrm{KDF}(K, Counter), \qquad (۹)$$
$$Counter = Counter + 1.$$

در پایان مرحله کدگشایی، آلیس $(Ver, Ttag)$ را برای باب ارسال می‌کند.

۵-۳- صحت‌سنجی کدگشایی

باب بعد از دریافت پیام $(Ver, Ttag)$، صحت آن را با استفاده از الگوریتم VToeplitz بیان شده در جدول (۹) بررسی می‌کند. اگر الگوریتم VToeplitz خروجی صفر را تولید کند، باب نتیجه می‌گیرد که پیام $(Ver, Ttag)$ توسط ایو دستکاری شده است. از سوی دیگر، اگر الگوریتم VToeplitz خروجی یک را تولید کند، باب نتیجه می‌گیرد که پیام توسط ایو دستکاری نشده است.

جدول (۹). گام اول صحت‌سنجی کدگشایی (الگوریتم VToeplitz). در این الگوریتم، پیام حاوی شماره بلوک‌های تایید شده آلیس توسط باب احراز اصالت می‌شود.

ورودی الگوریتم: $(Ver, Ttag)$، ماتریس T_s و بردار دوتایی r

[1] Toeplitz

خروجی الگوریتم: $v \in \{0,1\}$
۱- $m_{au} \leftarrow \text{Ver}$،
۲- $\alpha \leftarrow T_s\, m_{au} \oplus r$،
۳- اگر $Ttag = \alpha$ در این صورت $v \leftarrow 1$، اگر $Ttag \neq \alpha$ در این صورت $v \leftarrow 0$،
۴- v را به‌عنوان خروجی تولید کن.

باب بعد از اجرای الگوریتم VToeplitz، الگوریتم VerifiedBlocks2 را مطابق جدول (۱۰) اجرا می‌کند. این الگوریتم در صورتی که $v = 1$ باشد، کلید تأیید شده K_{Ver} را به عنوان خروجی تولید می‌کند. از سوی دیگر، اگر $v = 0$ باشد الگوریتم VerifiedBlocks2 خروجی Abort را تولید می‌کند که نشان می‌دهد ایو پیام $(Ver, Ttag)$ را دستکاری کرده است و آلیس و باب باید اجرای پروتکل را قطع کنند.

جدول (۱۰). گام دوم صحت‌سنجی کدگشایی (الگوریتم VerifiedBlocks2). در این الگوریتم، کلید تایید شده باب تولید می‌شود.

ورودی الگوریتم: v، K^B_{sift} و مجموعه Ver خروجی الگوریتم: K_{Ver} و $Res \in \{Abort, Continue\}$
۱- $K_{Ver} \leftarrow [\,]$،
۲- اگر $v = 0$ در این صورت $Res \leftarrow Abort$ و $K_{Ver} \leftarrow \emptyset$،
۳- اگر $v = 1$ در این صورت مراحل زیر را انجام بده:
۳-۱- برای هر $i \in \{1, \dots, l\}$ اگر $i \in Ver$ در این صورت $K_{Ver} \leftarrow K_{Ver} \| K^B_i$،
۳-۲- $Res \leftarrow Continue$،
۴- (K_{Ver}, Res) را به عنوان خروجی تولید کن.

باب بعد از اجرای الگوریتم VerifiedBlocks2، پیام Res را برای آلیس ارسال می‌کند. لازم به ذکر است که باب در کنار پیام Res، تگ $\text{PH}(Res, subk)$ را نیز جهت احراز اصالت برای آلیس ارسال می‌کند. در این صورت، اگر ایو پیام Res را دستکاری کند آلیس متوجه دستکاری ایو می‌شود.

در ادامه، آلیس و باب به طور مجزا کنارگذاشتن (Res = Abort) یا ادامه (Res = Continue) پروتکل را بررسی می‌کنند. درصورتی‌که Res = Abort باشد، هر دو پروتکل را قطع و یک پروتکل جدید را شروع می‌کنند. $Res = Abort$ به این معناست که ایو توانسته است پیام‌های تبادل‌یافته حین اجرای پروتکل را دستکاری کند. اگر Res = Continue باشد، در این صورت آلیس و باب پروتکل را ادامه می‌دهند. Res = Continue به این معناست که ایو نتوانسته است پیام‌های تبادل‌یافته حین اجرای پروتکل را دستکاری کند. در این حالت، آلیس و باب هر دو کلید K_{Ver} را در اختیار خواهند شد.

۵-۴- تخمین پارامتر

در هر سامانه QKD تجربی، به طور طبیعی و صرف‌نظر از مداخله ایو، خطاهایی رخ می‌دهد که ناشی از وجود نقص در قطعات سامانه مانند منبع نوری، کانال انتقال و یا آشکارسازها است. این منابع خطا در کمیتی که اصطلاحاً نرخ خطای بیت کوانتومی[1] (QBER) نامیده می‌شود، مطالعه و بررسی می‌شوند. در صورت حضور و دست‌کاری مؤثر ایو، این کمیت از یک مقدار بحرانی $QBER_{crit}$ (بیشینه نرخ خطای قابل تحمل توسط سامانه) فراتر رفته، امکان دستیابی به کلید امن از بین می‌رود. هدف از اجرای تخمین پارامتر در پروتکل پساپردازش، محاسبه تخمینی از مقدار QBER است. این تخمین با استفاده از رابطه زیر قابل محاسبه است [۱۲]:

$$QBER_{est} = N^{-1}\left(\sum_{i \in Ver} QBER_i + \frac{|\overline{Ver}|}{2}\right), \qquad (۱۰)$$

که در آن، $QBER_i$ احتمال تغییر یک بیت در بلوک i-ام، $\overline{Ver}$ مجموعه اندیس‌های بلوک‌های تأیید نشده و N تعداد بلوک‌هایی است که اصلاح خطا حین کدگشایی بر روی آن‌ها اعمال شده است. اگر یکی از بلوک‌ها متعلق به مجموعه $\overline{Ver}$ باشد در این صورت ممکن است بیت‌های آن با احتمال $\frac{1}{2}$ تغییر کند. بعد از محاسبه مقدار $QBER_{est}$، آلیس الگوریتم QBERTest را مطابق جدول (۱۱) اجرا می‌کند. این الگوریتم مقدار $QBER_{est}$ را به عنوان ورودی می‌گیرد و در صورتی که $QBER_{est} > QBER_{crit}$ باشد خروجی $Abort$ و در غیر این صورت خروجی $Continue$ را تولید می‌کند. خروجی $Abort$ نشان‌دهنده نشتی اطلاعات است و بیان می‌کند که ایو توانسته است به نحوی در ارتباطات بین آلیس و باب دستکاری کند و اطلاعات قابل توجهی از کلید K_{Ver} بدست آورد.

جدول (۱۱). تخمین پارامتر (الگوریتم QBERTest). در این الگوریتم، حضور احتمالی ایو در فرایند انتقال کوانتومی کشف می‌شود.

ورودی الگوریتم: مقادیر $QBER_{est}$ و $QBER_{crit}$ خروجی الگوریتم: $Res \in \{Abort, Continue\}$
۱- اگر $QBER_{est} > QBER_{crit}$ در این صورت $Res \leftarrow Abort$، در غیر این صورت $Res \leftarrow Continue$،
۲- Res را به عنوان خروجی تولید کن.

[1] Quantum Bit Error Rate (QBER)

بعد از اجرای الگوریتم QBERTest و تولید خروجی Res، آلیس Res را برای باب ارسال می‌کند. در کنار پیام Res، آلیس تگ $\mathrm{PH}(Res, subk)$ را نیز برای باب ارسال می‌کند تا در صورت دستکاری پیام توسط ایو، باب متوجه آن شود. مشابه آنچه در انتهای صحت‌سنجی کدگشایی گفته شد، در اینجا نیز نهایتاً آلیس و باب به طور مجزا بررسی می‌کنند که $\mathrm{Res} = \mathrm{Abort}$ است یا $\mathrm{Res} = \mathrm{Continue}$. در صورت که $\mathrm{Res} = \mathrm{Abort}$ باشد هر دو پروتکل را قطع و یک پروتکل جدید را شروع می‌کند. $Res = Abort$ به این معناست که اطلاعاتی از کلید K_{Ver} نشت پیدا کرده است و ایو به نحوی توانسته است اطلاعاتی از این کلید بدست بیاورد. اگر $\mathrm{QRes} = \mathrm{Continue}$ باشد در این صورت آلیس و باب پروتکل را ادامه می‌دهند. $\mathrm{QRes} = \mathrm{Continue}$ به این معناست نشتی اطلاعات از کلید K_{Ver} صورت نگرفته است و ایو اطلاعاتی از کلید در اختیار ندارد.

۶- مرحله تقویت محرمانگی در پروتکل پیشنهادی

با اجرای موفق مراحل بیان شده، آلیس و باب به کلید مشترک K_{Ver} دست پیدا خواهند کرد، ولی هنوز این احتمال وجود دارد که ایو بخشی از کلید K_{Ver} را بدست آورده باشد. هدف از اجرای مرحله تقویت محرمانگی، کاهش اطلاعات ایو از کلید K_{Ver} است به گونه‌ای که بتوان از اطلاعات باقی‌مانده ایو از کلید نهایی صرف نظر کرد و آن را کاملاً امن در نظر گرفت. بعد از اجرای این مرحله، آلیس و باب به کلید نهایی K_{sec} با طول l_{sec} دست پیدا خواهند کرد، به نحوی که ایو با احتمال بسیار ناچیزی ممکن است اطلاعاتی از برخی بیت‌های K_{sec} در اختیار داشته باشد. در این مرحله، آلیس ابتدا با استفاده از رابطه زیر مقدار l_{sec} را محاسبه می‌کند [۱۲]:

$$l_{sec} = \hat{k}_1\, l_{Ver}\big(1 - h(\hat{e}_1)\big) - leak_{EC} - 5\log_2\left(\frac{1}{\epsilon_{PA}}\right), \quad (۱۱)$$

که در آن، $\hat{k}_1$ کسر پالس‌های تک‌فوتونی، l_{Ver} طول کلید K_{Ver}، $\hat{e}_1$ تخمینی از QBER مربوط به پالس‌های تک‌فوتونی، و $h(.)$ تابع آنتروپی شانون (آنتروپی دوتایی) است. ϵ_{PA} نیز یک پارامتر امنیتی است که بهتر است کمتر از 10^{-12} در نظر گرفته شود. $leak_{EC}$ اطلاعات احتمالی فاش شده حین مرحله کدگذاری و کدگشایی است که نهایتاً برابر با طول کدکلمه‌ها و تگ‌های ارسال شده حین اجرای پروتکل می‌باشد.

بعد از محاسبه l_{sec}، آلیس یک ماتریس توپلیتز T از اندازه $l_{sec} \times l_{Ver}$ را با استفاده از رشته بیت محرمانه $(s_{1-l_{sec}}, s_{1-l_{sec+1}}, \dots, s_{l_{Ver-1}})$ تشکیل می‌دهد. آلیس (l_{sec}, l_{Ver}) را به عنوان پارامترهای ماتریس توپلیتز، از طریق کانال عمومی برای باب ارسال می‌کند. توجه شود که در اینجا نیز تگ $\mathrm{PH}(l_{sec} \| l_{Ver}, subk)$ به همراه (l_{sec}, l_{Ver}) برای باب ارسال می‌شود، تا باب بتواند پیام دریافت شده را احراز اصالت کند. آلیس و باب می‌توانند رشته بیت محرمانه $(s_{1-l_{sec}}, s_{1-l_{sec+1}}, \dots, s_{l_{Ver-1}})$ را با استفاده از تابع تولید کلید $\mathrm{KDF}(K, Counter)$ تولید کنند. بعد از تولید ماتریس توپلیتز T، کلید محرمانه K_{sec} با استفاده از رابطه زیر قابل محاسبه است:

$$K_{sec} = T\, K_{Ver}. \quad (۱۲)$$

همانند بخش ۵-۲، در اینجا نیز برای داشتن امنیت کافی بهتر است $l_h \geq 64$ باشد. نهایتاً مناسب است که آلیس رابطه $\frac{l_h}{l_{sec}} < \%15$ را بررسی کند، تا مطمئن شود که کلید K_{sec} دارای طول مناسبی است.

خلاصه‌ای از مجموعه فرایندها، مراحل و الگوریتم‌های مختلف به‌کاررفته در پروتکل پساپردازش پیشنهادی و نیز اهداف و نحوه اجرای هر یک از آن‌ها در جدول (۱۲) آورده شده است.

۷- نتیجه‌گیری

در این مقاله، پس از تشریح الگوریتم عمومی سامانه‌های QKD، یک پروتکل پساپردازش بومی معرفی شد. جزئیات مربوط به فرایندهای غربال‌گری، اصلاح خطا و تقویت محرمانگی این پروتکل و همچنین زنجیره الگوریتم‌های مختلف مورد نیاز هر فرایند، به‌صورت ساده بیان شد. ورودی‌ها و خروجی‌های هر الگوریتم و ریز دستورات مورد نیاز برای اجرای آن بیان گردید. پروتکل پیشنهادی کاملاً قابلیت استفاده در پساپردازش سامانه‌های QKD تجربی و آزمایشگاهی را دارد و در صورت پیاده‌سازی توسط یک‌زبان برنامه‌نویسی توانمند و برخورداری از یک رابط گرافیکی کاربر مناسب، می‌تواند در سامانه‌های QKD صنعتی و تجاری نیز به کار رود.

جدول (۱۲). فرایندها، مراحل و الگوریتم‌های مختلف پروتکل پساپردازش پیشنهادی به همراه اهداف و نحوه اجرای هر کدام از آن‌ها. عملیات مربوط به آلیس با رنگ تیره مشخص شده‌اند.

نام فرایند	نام مرحله	نام الگوریتم	هدف الگوریتم	توضیحات
غربال‌گری	گام اول غربال‌گری آلیس	SiftA1	ارسال پایه‌های آماده‌سازی آلیس برای باب	آلیس پایه‌های آماده‌سازی خود را به همراه یک تگ احراز اصالت برای باب ارسال می‌کند.
	غربال‌گری باب	SiftB	تولید کلید غربال شده باب	باب پس از احراز اصالت پیام آلیس، پایه‌های اندازه‌گیری خود را با پایه‌های آلیس مقایسه و شماره پالس‌هایی که اولاً آن‌ها را دریافت کرده، ثانیاً پایه‌های اندازه‌گیری او و آماده‌سازی آلیس برای آن‌ها مشابه بوده است را به همراه یک تگ احراز اصالت برای آلیس ارسال می‌کند. بیت‌های حاصل از اندازه‌گیری با پایه‌های مشابه، کلید غربال شده باب را تشکیل می‌دهند.
	گام دوم غربال‌گری آلیس	SiftA2	تولید کلید غربال شده آلیس	آلیس پس از احراز اصالت پیام باب، با استفاده از آن کلید غربال شده خود را از کلید خامی که در اختیار دارد تولید می‌کند.
اصلاح خطا	کدگذاری	Encoding	کدگذاری کلید غربال شده باب	باب کلید غربال شده خود را با استفاده از الگوریتم کدگذاری LDPC کدگذاری کرده، با کمک یک سیستم رمز یک‌بارمصرف رمز می‌کند.
		PH	تولید تگ احراز اصالت برای پیام حاوی کلید غربال شده کدگذاری شده باب	باب تابع درهمساز PH را بر روی هر کدام از بلوک‌های کلید غربال شده خود اعمال و یک تگ احراز اصالت را برای آن تولید می‌کند. سپس پیام کدگذاری شده باب به همراه تگ احراز اصالت آن برای آلیس ارسال می‌شود.
	کدگشایی	Decoding	کدگشایی پیام کدگذاری شده باب	آلیس بعد از دریافت پیام کدگذاری شده باب، پیام کدگذاری شده باب را کدگشایی می‌کند.
		Corr	مقایسه پیام کدگشایی شده و کلید غربال شده آلیس	آلیس پیام کدگشایی شده و کلید غربال شده خود را با هم مقایسه و یکسان بودن بلوک به بلوک آن‌ها را بررسی می‌کند. رشته بیت Corr از کنار هم قراردادن بلوک‌های مشابه ساخته می‌شود.
		VerifiedBlocks1	تولید کلید تأیید شده آلیس	آلیس با استفاده از تگ احراز اصالت باب، بلوک‌های Corr را احراز اصالت (تأیید) می‌کند. کلید تأیید شده از کنار هم قراردادن بلوک‌های تأیید شده ساخته می‌شود. آلیس شماره بلوک‌های تأیید شده را به همراه یک تگ احراز اصالت برای باب می‌فرستد.
	صحت‌سنجی کدگشایی	VToeplitz	احراز اصالت پیام حاوی شماره بلوک‌های تأیید شده آلیس	باب اصالت پیام حاوی شماره بلوک‌های تأیید شده آلیس را به‌وسیله تگ احراز اصالت ارسالی او ارزیابی می‌کند. اگر احراز اصالت موفقیت‌آمیز نباشد، نتیجه می‌گیرد که ایو پیام ارسالی آلیس را تغییر داده است.
		VerifiedBlocks2	تولید کلید تأیید شده باب	باب با استفاده از بلوک‌های تأیید شده آلیس، بلوک‌هایی از کلید غربال شده خود را که درست کدگشایی شده‌اند مشخص می‌کند. کلید تأیید شده باب از کنار هم قراردادن بلوک‌های مشخص‌شده به دست می‌آید. باب موفقیت یا شکست عملیات تولید کلید تأیید شده خود را به همراه یک تگ احراز اصالت برای آلیس ارسال می‌کند.
	تخمین پارامتر	QBERTest	کشف حضور احتمالی ایو در فرایند انتقال کوانتومی	آلیس پس از محاسبه QBER سامانه، میزان آن را با یک مقدار بحرانی از قبل تعیین شده ($QBER_{crit}$) مقایسه می‌کند. اگر $QBER_{est} > QBER_{crit}$، آلیس "شکست پروتکل" و در غیر این صورت، "ادامه پروتکل" را به همراه یک تگ احراز اصالت برای باب ارسال می‌کند.
تقویت محرمانگی			کاهش اطلاعات ایو از کلید تأیید شده و تولید کلید امن نهایی	آلیس ابتدا طول کلید امن نهایی را محاسبه می‌کند. سپس پارامترهای ماتریس توپلیتز با اندازه مناسب را تشکیل داده، به همراه یک تگ احراز اصالت برای باب ارسال می‌کند. کلید امن نهایی آلیس از ضرب ماتریس توپلیتز در کلید تأیید شده او به دست می‌آید.
				باب پس از احراز اصالت پیام آلیس، با ضرب ماتریس توپلیتز در کلید تأیید شده خود، کلید امن نهایی را به دست می‌آورد.

۸- مراجع

[1] C. H. Bennett, F. Bessette, G. Brassard, L. Salvail, and J. Smolin, "Experimental quantum cryptography," Journal of Cryptology, vol. 5, pp. 3–28, 1992.

[2] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," Theor. Comput. Sci., vol. 560, pp. 7–11, 2014.

[3] A. K. Ekert, "Quantum cryptography based on Bell's theorem," Phys. Rev. Lett., vol. 67, pp. 661–663, 1991.

[4] A. Aghanian, S. N. Doustimotlagh, "Generalized Version of the BB84 QKD Protocol with n Polarization Bases and Unequal Probabilities, " Journal of Electronical & Cyber Defence, vol. 9, No.1, 2020 (In Persian).https://dor.isc.ac/dor/DOR:20.1001.1.23224347.1400.9.1.10.7

[5] V. Zapatero, W. Wang, and M. Curty, "A fully passive transmitter for decoy-state quantum key distribution," Quantum Sci. Technol., vol. 8, p. 025014, 2023.

[6] S. Dong et al., "Decoy state semi-quantum key distribution," EPJ Quant. Technol., vol. 10, 2023.

[7] A. M. Toonabi, M. D. Darareh, and S. Janbaz, "A two-dimensional quantum key distribution protocol based on polarization-phase encoding," Int. J. Quantum Inf., vol. 17, p. 1950058, 2019.

[8] A. M. Toonabi, M. D. Darareh, and S. Janbaz, "High-dimensional quantum key distribution using polarization-phase encoding: security analysis," Int. J. Quantum Inf., vol. 18, p. 2050031, 2020.

[9] G. S. Vernam, "Cipher Printing Telegraph Systems For Secret Wire and Radio Telegraphic Communications," Transactions of the American Institute of Electrical Engineers, vol. XLV, pp. 295–301, 1926.

[10] S. J. Jonson, "Low-density parity-check codes," Iterative Error Correction, pp. 34–74, 2009.

[11] X. –Y. Hu, E. Eleftheriou, D.-M. Arnold, and A. Dholakia, "Efficient implementations of the sum-product algorithm for decoding LDPC codes," GLOBECOM'01. IEEE Global Telecommunications Conference, 2001.

[12] E. Kiktenko, A. Trushechkin, Y. Kurochkin, and A. Fedorov, "Post-processing procedure for industrial quantum key distribution systems," Journal of Physics: Conference Series, vol. 741, p. 012081, 2016.

[13] Q. Li, D. Le, X. Wu, X. Niu, and H. Guo, "Efficient bit sifting scheme of post-processing in quantum key distribution," Quantum Information Processing, vol. 14, pp. 3785–3811, 2015.

[14] E. Diamanti, "Security and implementation of differential phase shift quantum key distribution systems," Doctoral Dissertation, Stanford University, 2006.

[15] A. M. Toonabi, M. D. Darareh, "Introducing a decoy-state version of the high-dimensional polarization-phase (PoP) quantum key distribution protocol and explaining its implementation," Journal of Electronical & Cyber Defence, vol. 12, No.2,2024 (In Persian). https://dor.isc.ac/dor/20.1001.1.23224347.1403.12.2.8.8.

[16] T. Krovetz and P. Rogaway, "Fast Universal Hashing with Small Keys and No Preprocessing: The PolyR Construction," Information Security and Cryptology — ICISC 2000, pp. 73–89, 2001.

[17] H. Krawczyk, "New Hash Functions for Message Authentication," Lecture Notes in Computer Science, pp. 301–310, 1995.