چهارمین کنفرانس ملی ایده های نوین مهندسی برق
۲۰و۲۱ آبان ماه ۱۳۹۴ ـ دانشگاه آزاد اسلامی واحد اصفهان (خوراسگان)

تحلیل امنیت دو پروتکل تصدیق RFID فوق سبک وزن *LMAP* و M^2AP

محسن شریفی[۱]، سید مهدی سجادیه[۲]
[۱]کارشناسی ارشد دانشگاه آزاد اسلامی واحد خوراسگان، Mohsen mola1370@yahoo.com
[۲] استادیار دانشگاه آزاد اسلامی واحد خوراسگان

چکیده

در این مقاله، آسیب پذیری امنیتی دو پروتکل تصدیق متقابل *RFID* فوق سبک وزن: *LMAP* و M^2AP را که اخیرا توسط پریس-لوپز پیشنهاد شده مورد تحلیل قرار داده می شود. سپس دو حمله موثر به نام های حمله عدم همگام سازی و حمله کاملا فاش کننده در مقابل پروتکل های آنها تشخیص داده خواهد شد. حمله اول می تواند همگام سازی بین قرائت کننده *RFID* و تگ را در اجرای یک پروتکل منفرد دچار اختلال کنند به گونه ای که آنها نتوانند یکدیگر را در اجرای هر پروتکل پیرو تصدیق کنند. حمله بعدی می تواند همه اطلاعات سری ذخیره شده در یک تگ را با بازپرسی چند زمانه تگ فاش کند. بنابراین تگ را کاملا خالی می کند. علاوه بر این ما اقدامات بالقوه را برای بهبود امنیت پروتکل های فوق خاطر نشان می کنیم.

۱ـ مقدمه

سیستمهای تشخیص فرکانس رادیویی (RFID) به صورت فزاینده ای در بسیاری از کاربردها گسترش یافته اند ولی استفاده فراگیر از آنها اساسا به دلیل نگرانیهای امنیتی و اختفا محدود شده است. چون تگهای RFID کلا کم هزینه و دارای منابع بسیار محدود هستند، اصول امنیت معمول نمی تواند به خوبی یکی شود. اما هنگامی که آنها در یک محیط فراگیر گسترش می یابند، جایی که تهدیدها غیرعادی نیستند، مسائل امنیت و اختفا باید قبل از بکارگیری وسیع مدیریت شوند.

در این مقاله، ما امنیت دو پروتکل تصدیق متقابل RFID فوق سبک وزن یعنی LMAP و M^2AP را که اخیرا توسط پریس-لوپز و سایرین پیشنهاد شده، مورد تحلیل قرار می دهیم. متفاوت از اکثر راه حلهای موجود که از اصول اختفای کلاسیک استفاده می کنند، این دو پروتکل بسیار سبک وزن هستند، زیرا فقط عملیات منطقی ساده ای را برای دستیابی به تصدیق متقابل بین قرائت کننده RFID و تگها استفاده می کنند. متعاقبا، فقط حدود ۳۰۰ تگ برای اجرای چنین ارتباطی بین RFID و تگ مورد نیاز است. علاوه بر این ادعا می شود که پروتکل های LMAP و هم M^2AP هر دو در مفهوم "پیشگیری از حمله" و "مقاوم بودن در مقابل جعل" ایمن هستند. به هر حال، ما برخی آسیب پذیری ها را در این دو پروتکل تشخیص می دهیم. خصوصا ابتدا نشان می دهیم که پروتکل ها متحمل حمله عدم همزمانی می شوند. این حمله فقط با استراق سمع یک پروتکل منفرد موثر بوده و بنابراین می تواند "همزمانی" بین پایگاه داده[۱] و تگ را از بین ببرد. بنابراین تگ نمی تواند بیش از این توسط پایگاه داده معتبر شناخته شود. سپس ما یک حمله جدی تر – حمله کاملا فاش کننده- را ارائه می کنیم. با اثر متقابل بین قرائت کننده (زمانهای O(1)) و تگ (زمانهای O(m)) این حمله یک مهاجم را قادر می سازد که ID (شناسه) تگ را به همراه سایر اطلاعات سری ذخیره شده در تگ کشف کند. بنابراین همه

ویژگیهای امنیتی ذکر شده توسط پروتکل های بالا از بین می رود. نهایتا برای دفاع در مقابل حملات فرق، ما چندین اقدام متقابل بالقوه را پیشنهاد می کنیم. یکی از آنها خاصیت بی قاعدگی پروتکل های اصلی را مدیریت می کند که می تواند با افزودن اطلاعات وضعیت در پروتکل ها بهبود یابد. به عنوان نتیجه، فضای حافظه اضافی (حدود ۴۰ درصد) برای اجرای چنین تگی مورد نیاز است.

بقیه این مقاله به صورت زیر سازمان یافته است. بخش ۲ به طور کلی کارهای مرتبط در مورد امنیت RFID و مسائل امنیتی را مرور می کند. سپس ما LMAP و M²AP را در بخش ۳ مرور می کنیم و آسیب پذیری های آنها را در بخش ۴ مورد تحلیل قرار می دهیم. بخش ۵ چندین اقدام متقابل را خاطر نشان می کند. در نهایت، از مقاله نتیجه گیری می کنیم.

۲ـ مسائل امنیتی و اختفا در سیستم های RFID

۲,۱ مدل تهدید

ازدیاد تگهای RFID تایید می‌کند که فنـاوری فراگیر RFID مـی‌تواند خطرات غیرمنتظره ای به وجود آورد. عملا بیش از صدها مقاله تحقیقاتی در برخورد با مسائل امنیت و اختفای RFID منتشر شده است . برخی تحقیقات قبلی بر محدودیت های عملی بکارگیری RFID فرضهایی می گذارند: یعنی اولا در نظر می گیرند که کانال تگ به قرائت کننده خصوصی است، زیرا کانال انتشار برگشتی از تگ به قرائت کننده نسبت به کانال مستقیم محدوده کوتاهتری (مانند چند سانتیمتر) دارد. بنابراین یک مهاجم با محدوده نمی تواند پاسخی از تگ دریافت کند. و ثانیا برای یک مهاجم راحت نیست که در یک اجرای فعال خودش را بین یک قرائت کننده قانونی و یک تگ مخفی کند. بنابراین بین تگ و قرائت کننده هیچ مرد میانی (مهاجم) وجود ندارد. همچنین ثالثا به دلیل محیط اتصال بی سیم به اشتراک گذاشته شده، قطع یک پیام و تغییر پیام در فضا در زمان طبیعی ساده نیست. در حالی که این فرضیات در گسترش بسیاری از RFID های عملی برای ایجاد امنیت معقول بکار می روند، در مواجهه با تهدیدات پیش رو در محیط های گسترش صریح، محکم و کافی نیستند.

۳ـ مرور LMAP و M²AP

در پروتکل LMAP ، عملیات ساده ای مانند: XOR منطقی ($\oplus$)، OR منطقی ($\vee$)، AND منطقی ($\wedge$)، و حالت جمع 2^m (+) استفاده می شود. عملیات با ارزش مانند ضرب و محاسبات نویز (پارازیت) اصلا مورد نیاز نیستند و تولید عدد تصادفی فقط توسط قرائت کننده انجام می شود. این طرح اسامی شاخص (IDS ها) را استفاده می کند. یک اسم شاخص (طول ۹۶ بیت) شاخص یک جدول (یک سطر) است که در آن همه اطلاعات مربوط به تگ ذخیره می شود. هر تگ مرتبط با یک کلید است که به چهار قسمت ۹۶ بیتی (K=K1||K2||K3||K4) تقسیم می شود. چون IDS و کلید (K) باید بروز شوند، کلا ۴۸۰ بیت از حافظه قابل بازنویسی (EEPROM) را نیاز دارد. یک حافظه ROM نیز برای ذخیره عدد شناسایی ثابت ۹۶ بیت مورد نیاز است. پروتکل در جدول ۱ نشان داده شده است.

شناسایی تگ: قرائت کننده ← تگ: سلام	جایی که:
تصدیق متقابل LMAP: قرائت کننده← تگ: A||B||C تگ ← قرائت کننده:D	$A = IDS^{(n)}_{tag(i)} \oplus K1^{(n)}_{tag(i)} \oplus n1$ $B = (IDS^{(n)}_{tag(i)} \vee K2^{(n)}_{tag(i)}) + n1$ $C = IDS^{(n)}_{tag(i)} + K3^{(n)}_{tag(i)} + n2$ $D = (IDS^{(n)}_{tag(i)} + ID_{tag(i)}) \oplus n1 \oplus n2$
تصدیق متقابل M² AP: قرائت کننده ←تگ: A||B||C تگ ← قرائت کننده:D||E	جایی که A، C همان A، C موجود در LMAP هستند. $B = (IDS^{(n)}_{tag(i)} \wedge K2^{(n)}_{tag(i)}) \vee n1$ $D = (IDS^{(n)}_{tag(i)} \vee ID_{tag(i)}) \wedge n2$ $E = (IDS^{(n)}_{tag(i)} + ID_{tag(i)}) \oplus n1$

جدول ۱. پروتکل LMAP و M²AP

این پروتکل سه مرحله اصلی دارد: شناسایی تگ، تصدیق متقابل، بروز رسانی اسم شاخص و بروز رسانی کلید.

-شناسایی تگ: قرائت کننده یک پیام سلام به تگ ارسال می کند که توسط اسم شاخص (IDS) کنونی خود پاسخ خواهد داد. توسط این IDS، فقط یک قرائت کننده معتبر می تواند به کلید سری متناظر تگ (K+K1||K2||K3||K4) که برای انجام مرحله شناسایی بعدی ضروری است، دست یابد.

تصدیق متقابل: ابتدا قرائت کننده دو عدد تصادفی n1 و n2 تولید می کند. با n1، n2 و کلیدهای فرعی K1، K2 و K3، قرائت کننده پیامهای فرعی A، B و C را تولید می کند و سپس آنها را به تگ ارسال می کند. با پیام های فرعی A و B، تگ می تواند قرائت کننده را تصدیق (تایید) کند و به n1 دست یابد. وقتی برای بار اول قرائت کننده تصدیق شد، تگ می تواند عدد تصادفی n2 را از پیام فرعی C به دست آورد و سپس پیام پاسخ D را تولید می کند. در این روش، شناسع ثابت تگ به صورت ایمن به قرائت کننده ارسال می شود. اگر قرائت کننده بتواند یک ID معتبر از پیام D بیابد، تگ با موفقیت تایید می شود. توجه کنید که اعداد تصادفی n1 و n2 نیز برای بروز رسانی اسم شاخص و کلید استفاده می شوند.

بروز رسانی اسم شاخص و کلید: پس از اینکه قرائت کننده و تگ یکدیگر را تایید کردند، آنها بروز رسانی اسم شاخص و کلید را با معادلات زیر انجام می دهند.

$$IDS_{tag(i)}^{(n+1)} = (IDS_{tag(i)}^{(n)} + (n2 \oplus K4_{tag(i)}^{(n)})) \oplus ID_{tag(i)}$$
$$K1_{tag(i)}^{(n+1)} = K1_{tag(i)}^{(n)} \oplus n2 \oplus (K3_{tag(i)}^{(n)} + ID_{tag(i)})$$
$$K2_{tag(i)}^{(n+1)} = K2_{tag(i)}^{(n)} \oplus n2 \oplus (K4_{tag(i)}^{(n)} + ID_{tag(i)})$$
$$K3_{tag(i)}^{(n+1)} = (K3_{tag(i)}^{(n)} \oplus n1) + (K1_{tag(i)}^{(n)} \oplus ID_{tag(i)})$$
$$K4_{tag(i)}^{(n+1)} = (K4_{tag(i)}^{(n)} \oplus n1) + (K2_{tag(i)}^{(n)} \oplus ID_{tag(i)})$$

LMAP [۱۴] یک پروتکل خواهر به نام M^2AP [۱۵] دارد که یک پروتکل تصدیق متقابل RFID سبک وزن بسیار مشابه است. معادله بروز رسانی اسم شاخص در M^2AP با اندکی تفاوت نسبت به LMAP به

$$IDS_{tag(i)}^{(n+1)} = \left(IDS_{tag(i)}^{(n)} + (n2 \oplus n1)\right) \oplus ID_{tag(i)}$$

تغییر می یابد. همه عملیات بروز رسانی کلید مشابه LMAP است. جدول ۱ M^2AP را نیز توصیف می کند.

مولفان برخی تحلیلهای امنیت را ارائه نمودن و اظهار کردند که LMAP و M^2AP هر دو در مقابل موارد: بی نامی تگ، تصدیق متقابل، جلوگیری از حمله مرد میانی، ممانعت از حمله پاسخ، مقاومت در برابر جعل ایمن هستند. در بخش بعدی، ما حملات موثر را که می تواند پروتکل های بالا را نقض کرده و کاستی های اظهارات آنها را نشان دهد، تعیین می کنیم.

۴ـ آسیب پذیری LMAP و M^2AP

قبل از همه، ما اظهار می کنیم که پروتکل های بالا در مفهوم پروتکل های نهفته مقاوم نیستند، زیرا اگر D توسط یک قرائت کننده قانونی به درستی دریافت شود یا تغییر کند، تگ نمی تواند تشخیص دهد. اگر D با موفقیت دریافت نشود یا تغییر نکند، قرائت کننده حافظه مرتبط با تگ خود را بروز نمی کند، در حالی که تگ حافظه اش را بروز می کند چون اکنون قرائت کننده را تایید کرده است. به وضوح حافظه تگ و قرائت کننده همزمان نیستند. اما این موضوع بیشتر در مورد یک مسئله فرضی و نه به عنوان یک مسئله امنیتی جدی است. به منظور تعیین مفهومی این مطلب، فرض می کنیم که یک پیام انجام برای نشان دادن اجرای موفق یک پروتکل فرستاده شده است. این پیام انجام، عملیات بروز رسانی را در هر دو سمت قرائت کننده و تگ مقدور می سازد. همه حملات پیرو فرض می کنند که پروتکل ها، پیام انجام فوق را برای اجرای بروز رسانی دارند. پیرو این موضوع، ما مسائل امنیتی LMAP را همانند M^2AP ارائه می کنیم.

۴,۱ حمله عدم همزمانی

به منظور ایجاد امنیت برای تگ RFID، اکثر پروتکل های تصدیق RFID، یک ID تگ را پس از تکمیل پروتکل موفق بروز رسانی می کنند. نوعا پایگاه داده باید ID تگ را متعاقبا بروز کند تا یک قرائت کننده قانونی بتواند همچنان تگ را تصدیق کند. بنابراین همزمان سازی اطلاعات سری بین پایگاه داده و تگ برای اجرای پروتکل های موفق متعاقب آنها بسیار مهم است. یک پروتکل نقض شده همچنانکه در بالا بحث شد، ممکن است پروتکل ها را به صورا ناتمام رها کند و باعث عدم همزمانی در هر دو سمت شود. علاوه بر این، یک حمله خواسته مانند حمله عدم همزمانی که در زیر معرفی می شود، نیز می تواند پروتکل های تصدیق را نقض کند.

حمله ۱: تغییر پیام C. اکنون ساده ترین حمله عدم همزمان سازی را ارائه می کنیم: بدون دانش قبلی در مورد

پروتکل قبلی، یک مرد میانی نمی تواند در ابتدا بر روی پروتکل در حال اجرا استراق سمع کند و سپس A||B||C را به A||B||C' تغییر دهد که[I]$_0$= [000…001] C'=C$\oplus$ است. (اولین ۹۵ بیت با ارزش I را صفر و کم ارزش ترین بیت را ۱ قرار دهید). به صورت مشابه مهاجم پاسخ D از تگ را به D'=D$\oplus$[I]$_0$ تغییر می دهد. این فرایند در جدول ۲ ترسیم شده است.

تصدیق متقابل LMAP: قرائت کننده تگ: A||B||C' تگ ← قرائت کننده: D'	جایی که: $n2' \leftarrow n2$ $C' = C \oplus [I]_0$ $D = (IDS^{(n)}_{tag(i)} + ID_{tag(i)}) \oplus n1 \oplus n2'$ $D' = D \oplus [I]_0$

جدول ۲. حمله عدم همزمانی در برابر LMAP

در سمت تگ، حمله بر روی اولین اجرای پروتکل متقابل: "شناسایی تگ" تاثیر نمی گذارد. اما در اجرای دوم، هنگامی که تگ پیام A||B||C' را دریافت می کند، تا زمانی که A و B حفظ می شوند، هنوز می تواند قرائت کننده را تصدیق کند. اما تگ عدد تصادفی اشتباه n2 ← n2'را می گیرد (که n2' به n2 بستگی دارد، اما بر طبق معادله ۴-۱، الزاما به عنوان تابعی از n2 تعریف نمی شود). تگ این مقدار را خواهد پذیرفت و پاسخ آن را براساس n2',D=$(IDS^{(n)}_{tag(i)} + ID_{tag(i)}) \oplus n1 \oplus n2'$ محاسبه می کند. در این ساده ترین حمله، اکنون مهاجم می تواند پاسخ D' را برای قرائت کننده ایجاد کند. اگر قرائت کننده مقدار D' را بپذیرد، می گوییم حمله موفق است؛ در غیر این صورت حمله شکست می خورد. اکنون نرخ موفقیت را به صورت زیر تحلیل می کنیم: عمل بر روی C عملا کم ارزش ترین بیت C [که به صورت [C]$_0$مشخص می شود) را تغییر می دهد:

$$[C]_0 = 1; \Rightarrow [C']_0 = 0; \rightarrow If \quad [n2]_0 = 0, HW(n2 \oplus n2') \geq 2 \quad \text{اگر} \quad (1)$$
$$\rightarrow If \quad [n2]_0 = 1, n2' = n2 \oplus [I]_0 \quad (2)$$
$$[C]_0 = 0; \Rightarrow [C']_0 = 1; \rightarrow If \quad [n2]_0 = 0, n2' = n2 \oplus [I]_0 \quad \text{اگر} \quad (3)$$
$$\rightarrow If \quad [n2]_0 = 1, HW[n2 \oplus n2'] \geq 2 \quad (4)$$

در اینجا، HW(a) وزن همینگ a است، بنابراین(b$\oplus$HW(a اختلاف تعداد بیتها بین a و b را مشخص می کند. توجه کنید که برای حالت های (۱) و (۲)، n2'=n2-1 است؛ و برای حالت های (۳) و (۴)، n2'=n2+1 است. برای حالت های (۲) و (۳)، قرائت کننده D' را خواهد پذیرفت، چون [I]$_0$=$(IDS^{(n)}_{tag(i)} + ID_{tag(i)}) \oplus n1 \oplus n2$ =$IDS^{(n)}_{tag(i)} + ID_{tag(i)}) \oplus n1 \oplus n2' \oplus [I]_0$ D'=D$\oplus$ است. برای حالات (۱) و (۴)، قرائت کننده به دلیل عدم تطابق موقعیت بیتهای متناظر(بیشتر از یکی) آن را نخواهد پذیرفت. فرض کنید که n2 به صورت تصادفی تولید می شود، نرخ موفقیت ۵۰ درصد در مورد ساده ترین حمله وجود دارد.

وقتی برای بار اول قرائت کننده مقدار را پذیرفت، قرائت کننده نیاز دارد که اطلاعات سری تگ را با زوج (n1,n2) بروز رسانی کند. به هر حال، تگ یک زوج دیگر (n1',n2') را برای بروز رسانی اطلاعات سری خود استفاده می کند. به عنوان مثال $IDS^{(n+1)}_{tag(i)} = (IDS^{(n)}_{tag(i)} + (n2' \oplus K4^{(n)}_{tag(i)})) \oplus ID_{tag(i)}$.

واضح است که عدم تطابقی در ذخیره اطلاعات سرس برای تگ و قرائت کننده وجود دارد (به جدول ۳ مراجعه کنید). با این خاتمه، ساده ترین حمله فرض می کند که فقط تغییر یک (کم ارزش ترین) بیت در پیام C وجود دارد. حمله موثر است زیرا برای دو تلاش یک بار موفق می شود. در حقیقت، ساده ترین حمله می تواند برای تغییر یک بیت منفرد از C در هر موقعیت i گسترش یابد، بنابراین می تواند یک حمله کلی با نرخ موفقیت مشابه (۵۰ درصد) باشد.

حمله عدم همزمان سازی تعمیم یافته: برای هر پروتکل LMAP در حال اجرا، یک مهاجم می تواند پیام C را قطع کند و هر بیت از C را تغییر دهد و C' را که به صورت C'=C$\oplus$ [I]$_j$ (0≤j≤95) است، ایجاد کند. سپس پیام جدید A||B||C' به تگ ارسال می شود. به محض دریافت یک پاسخ D از تگ، مهاجم آن را به D'=D$\oplus$ [I]$_j$ تغییر می دهد و آن را به قرائت کننده ارسال می کند. بر اساس تحلیل فوق، نرخ موفقیت حمله ۵۰ درصد است. یک حمله موفق می تواند وضعیت سری تگ را بر روی یک قرائت کننده تغیر دهد یا می تواند گفت این حمله قرائت کننده و تگ را دچار عدم همزمانی می کند.

حمله ۲: تغییر پیام A و B. علاوه بر این، حمله می تواند بر روی n1 نیز هدف گیری کند. در این حالت، مهاجم پیام A||B||C را قطع می کند و A'||B'||C را به تگ ارسال می کند که A'=A$\oplus$ [I]$_j$ و B'=B$\oplus$ [I]$_j$ است، یعنی ما بیت jام از A و B را تغییر می دهیم.

چون $A=IDS_{tag(i)}^{(n)} \oplus K1_{tag(i)}^{(n)} \oplus n1$، ما $n1'=n1 \oplus [I]_i$ را قرار می دهیم. برای B، به دست می آوریم

$$If \ [B]_j = 1; \Rightarrow [B']_j = 0; \rightarrow If \ [n1]_j = 0, HW(n1 \oplus n1') \geq 2 \quad (5)$$
$$\rightarrow If \ [n1]_j = 1, n1' = n1 \oplus [I]_j \quad (6)$$
$$If \ [B]_j = 0; \Rightarrow [B']_j = 1; \rightarrow If \ [n1]_j = 0, n1' = n1 \oplus [I]_j \quad (7)$$
$$\rightarrow If \ [n1]_j = 1, HW(n1 \oplus n1'] \geq 2 \quad (8)$$

که در آن، برای حالات (۵) و (۶)، $n1'=n1-2^j$ بوده؛ و برای حالات (۷) و (۸)، $n1'=n1+2^j$ است. برای حالات (۶) و (۷)، تگ، قرائت کننده را با پذیرش n1' تصدیق می کند. برای حالات (۱) و (۴)، تگ، قرائت کننده را تصدیق تصدیق نخواهد کرد. فرض کنید که n1 به صورت تصادفی تولید می شود، مهجم ۵۰ درصد شانس موفقیت دارد تا تگ را فریب دهد. فرض کنید که تگ پیام ساختگی (A',B') را بپذیرد، او پیام D را برای تکمیل پروتکل تولید می کند. مهاجم نیاز دارد که $D'=D \oplus [I]_j$ را با هر پاسخ معتبر از تگ به قرائت کننده ارسال کند. و این پیام D' توسط یک قرائت کننده با موفقیت بازبینی می شود. به محض یک حمله موفق، قرائت کننده و تگ هر دو می خواهند اطلاعات سری خود را بروز کنند. قرائت کننده با زوج (n1,n2) بروز رسانی می کند، در حالی که تگ (n1',n2) را استفاده می کند که باعث عدم تطبیق در اجرای بعدی پروتکل شناسایی می شود (به جدول ۳ مراجعه کنید).

ت**حلیل حمله**. در مقایسه با حمله ۱، که هدف در پروتکل جزئی است که قرائت کننده تگ را شناسایی می کند، حمله ۲ در فرایندی است که تگ قرائت کننده را شناسایی می کند. حمله فوق می تواند به حمله ۳ تعمیم یابد: اگر ما به صورت همزمان n1 و n2 را تغییر دهیم، نیازی به تغییر D نیست. در این حالت، مهاجم پیام را قطع می کند و A'||B'||C' را ارسال می کند.

شانس موفقیت حدود ۲۵ درصد است. تاثیرات در بروز رسانی قرائت کننده و تگ در جدول ۳ خلاصه شده است.

حملات	شانس موفقیت	حافظه قرائت کننده	حافظه تگ
حمله ۱	۵۰ درصد	$[IDS, K1, K2, K3, K4]$	$[IDS', K1', K2', K3, K4]$
حمله ۲	۵۰ درصد	$[IDS, K1, K2, K3, K4]$	$[IDS, K1, K2, K3', K4']$
حمله۳	۲۵درصد	$[IDS, K1, K2, K3, K4]$	$[IDS', K1', K2', K3', K4']$

جدول۳. حافظه های بروز شده در قرائت کننده و تگ پس از حملات

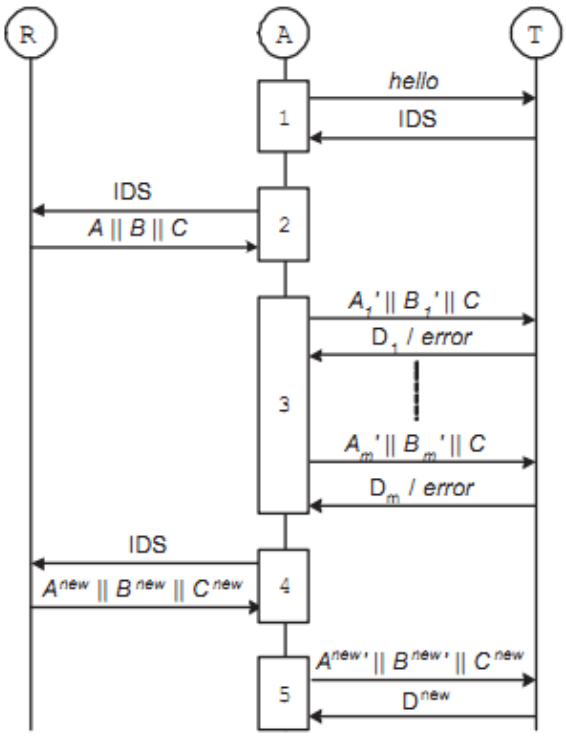
۴,۲ حمله کاملا افشا کننده

با داشتن حملات فوق، ما می توانیم ID اصلی یک تگ را که مهم تر است، افشا کنیم. فرض کنید که تگ هیچ حافظه ای برای اطلاعات وضعیت ندارد (بنابراین بدون وضعیت در نظر گرفته می شود)، اما یک قرائت کننده قانونی دارای وضعیت است (نظر به اینکه همه اطلاعات وضعیتی راجع به پروتکل در یک تگ خاص به خاطر سپرده می شود). این موضوع به این معنی است که ما می توانیم مکررا پروتکل ناکامل را به دفعات در سمت تگ اجرا کنیم. این فرض معقول است زیرا تگ باید به هر درخواست توسط قرائت کننده های قانونی یا غیرقانونی پاسخ دهد و اگر قرائت کننده پیام نهایی D را دریافت نکند، پروتکل کامل نیست.

این حمله در شکل ۱ نشان داده شده است. مرحله ۱، یک مهاجم یک قرائت کننده قانونی را جعل می کند و IDS کنونی از یک تگ را اتخاذ می کند. مرحله ۲، مهاجم با استفاده از این IDS معتبر، خود را به جای یک تگ جا می زند تا یک پیام معتبر A||B||C را از قرائت کننده قانونی به دست آورد. مرحله ۳، مهاجم سعی می کند که همه A'||B'||C های ممکن را به تگ ارسال کند که A' و B' به ترتیب با تغییر بیت jام A و B به دست می آید ($0 \leq j \leq 95$). براساس اینکه یک D درست یا یک پیام خطا دریافت شود (مهاجم نیاز ندارد که مقدار را بداند، یک نشان دهنده خطا برای مهاجم کافی است تا تصمیم خود را اتخاذ کند)، مهاجم نتیجه می گیرد که بیت jام از n1 برابر یا بیت jام از B است یا خیر. با این روش، با فقط ۹۶ تلاش، مهاجم می تواند مقدار کامل بیتهای n1 را به دست آورد. بنابراین مهاجم می تواند از A، B، IDS و n1 مقادیر K1 و K2 را محاسبه کند.

اکنون پارامترهای نامشخص n2، K3، K4 و ID هستند. به وضوح، ما می توانیم روش فوق را برای دستیابی به مقدار n2 برای m بار تعامل با قرائت کننده به کار ببریم. به هر حال، تلاشهای تکرار کننده توسط مهاجم به سادگی توسط یک قرائت کننده با وضعیت کامل شناسایی می شود و با محدود نمودن تعـاملات به یک ثـابت (به عنوان مثال بیشتـر از ۱۰) مقابله

مـی‌شود. با این فرض، ما باید راه دیگری را برای استخراج اطلاعات سری در پیش بگیریم. بنابراین در مرحله ۴، مهاجم وانمود می کند که یک تگ قانونی است و IDS را دوباره به قرائت کننده ها ارسال می کند (تعامل 2nd با قرائت کننده). قرائت کننده به صورت $A^{new}||B^{new}||C^{new}$ پاسخ می دهد. سپس، در مرحله ۵، مهاجم می تواند (با استفاده از پارامترهای معلوم کنونی IDS، K1 و K2) $n1^{new}=0$ قرار دهد و $A^{new'}||B^{new'}||C^{new}$ را به تگ ارسال کند. تگ با D^{new} پاسخ می دهد. توجه کنید که در مراحل فوق، کلا ۲ تعامل بین قرائت کننده و مهاجم و m+2 تعامل بین مهاجم و تگ وجود دارد.



شکل ۱. حمله کاملا افشا کننده

ت این جا مهاجم معادلات زیر را دارد:

$$C = (IDS_{tag(i)}^{(n)} + K3_{tag(i)}^{(n)}) + n2 \quad (9)$$

$$D = (IDS_{tag(i)}^{(n)} + ID_{tag(i)}) \oplus n1 \oplus n2 \quad (10)$$

$$C^{new} = (IDS_{tag(i)}^{(n)} + K3_{tag(i)}^{(n)}) + n2^{new} \quad (11)$$

$$D^{new} = (IDS_{tag(i)}^{(n)} + ID_{tag(i)}) \oplus n2^{new} \quad (12)$$

پس از آن، مهاجم می تواند $ID_{tag(i)}$ را به صورت زیر حل کند. ابتدا با حذف n2 از معادلات (۹) و (۱۰)، و $n2^{new}$ از معادلات (۱۱) و (۱۲)، ما معادلاتی با پارامترهای مجهول ID و K3 را می یابیم:

$$C - IDS_{tag(i)}^{(n)} - K3_{tag(i)}^{(n)} = (IDS_{tag(i)}^{(n)} + ID_{tag(i)}) \oplus n1 \oplus D \quad (13)$$

$$C^{new} - IDS_{tag(i)}^{(n)} - K3_{tag(i)}^{(n)} = (IDS_{tag(i)}^{(n)} + ID_{tag(i)}) \oplus D^{new} \quad (14)$$

علاوه بر این K3 را از دو معادله فوق حذف می کنیم و به دست می آوریم

$$C^{new} - C = (IDS_{tag(i)}^{(n)} + ID_{tag(i)}) \oplus D^{new} - (IDS_{tag(i)}^{(n)} + ID_{tag(i)}) \oplus n1 \oplus D$$

اکنون بحث می کنیم که چگونه $ID_{tag(i)}$ را از معادله فوق به دست آوریم. $c=C^{new}-C \mod 2^{96}$ ، $b=n1 \oplus D$ ، $a= D^{new}$ و $x=(IDS_{tag(i)}^{(n)} + ID_{tag(i)}) \mod 2^{96}$ قرار می دهیم. چون اکنون $IDS_{tag(i)}^{(n)}$ معلوم است، مسئله معادل با یافتن $x \in \{0,1\}^{96}$ برای (a,b,c) داده شده است به گونه ای که

$$x \oplus a = x \oplus b + c \mod 2^{96}. \quad (15)$$

برای حل x در معادله (۱۵)، ما فقط نیاز داریم توجه کنیم که با ارزش بیتهای x در محاسبه شامل کم ارزش ترین بیتها تاثیر ندارند. بنابراین می توانیم برای تعیین x از کم ارزش ترین بیتهایش به بیتهای با ارزش تر آن تلاش کنیم. به عنوان مثال، می توانیم ۹۶ بیت را به ۲۴ بخش تقسیم کنیم تا اینکه هر بخش ۴ بیت داشته باشد. پس از آن، منحصرا با جستجو می توانیم همه راه حلهای موجود برای اولین ۴ بیت کم ارزش x را بیابیم، سپس ۴ بیت کم ارزش بعدی x و امثالهم. این فرایند به دلیل رقم های نقلی ممکن در موقعیتهای همه بیتهای (4k+1)ام ($1 \le k \le 23$) بیشتر از $(2^{24}-1)$ مرتبه جستجوی انحصاری همه رشته های ۴ بیتی طول نمی کشد. این موضوع به این معنی است که چنین الگوریتم ساده ای می تواند توسط یک رایانه شخصی در عرض چند دقیقه انجام شود. عملا بکارگیری الگوریتم های موثر پیشنهاد شده در [۱۰]، معادله (۱۶) مجددا می تواند در پیچیدگی O(m) حل شود (m=96 در پروتکل ها). توجه کنید که با سه گانه (a,b,c) داده شده، ممکن است نتوان به صورت یکتا مقدار x را تعیین نمود. در این سناریو، مهاجم می تواند برای دست یابی به چند نمونه از معادله (۱۵)، با قرائت کننده چندین مرتبه[۲] تعامل کند. با تقسیم مجموعه راه حلهای این نمونه های مختلف، محدوده مقدار x می تواند به صورت

مشخصی باریک شود. علاوه بر این، چون $ID_{tag(i)}$ یک عدد تصادفی درست نیست اما فرمت ثابتی دارد، برخی بیتهای x تقریبا از قبل تعریف شده اند. با ترکیب این روشها، احتمال دارد که مقدار x به صورت یکتا با تعاملات کافی اما نه زیاد با قرائت کننده و تگ تعیین شود. وقتی برای بار اول مقدار x ثابت شد، حمله می تواند به راحتی بقیه اطلاعات سری (ID، K3، K4) ذخیره شده در تگ را استخراج کند. این امر حمله کاملا افشا کننده ما را در مقابل LMAP کامل می کند.

توجه کنید که حمله کاملا افشا کننده فوق در مقابل پروتکل LMAP می تواند برای حمله به پروتکل M^2AP نیز به کار رود. عملا، این حمله فقط نیاز به مراحل حمله از ۱ تا ۳ دارد که از آن می توانیم n1 را بیابیم. پس از آن با یک E معتبر، ID را مستقیما می یابیم. این امر ایجاب می کند که حماه کاملا افشا کننده در مقابل M^2AP موثرتر از حمله LMAP است زیرا مهاجم فقط یک تعامل با قرائت کننده و m+1 تعامل با تگ دارد.

۵ـ اقدامات متقابل

۵,۱ همزمان سازی مجدد

در حقیقت، در یک بسط ساده $LMAP^{+}-$ مولفان روشی را در مورد همزمان سازی مجدد بین قرائت کننده و تگ ذکر کرده اند. تگ حالتی مرتبط با پایگاه داده دارد: همزمان یا غیرقطعی. علاوه بر این، هر تگ به جای فقط یک سابقه، l+1 سابقه در پایگاه داده خواهد داشت. اولین سابقه، اسم شاخص واقعی (IDS) است و سایر سابقه ها اسامی شاخص بالقوه بعدی (IDS+1، IDS+2، ... IDS+l) است. پارامتر l توسط اندازه پایگاه داده مشخص می شود، بنابراین نمی تواند برای همه سابقه های ذخیره شده در پایگاه داده خیلی بزرگ باشد. گسترش می تواند به همزمان سازی مجدد برخی موقعیت های غیر همزمانی کمک کند. متاسفانه، این روش فقط می تواند بر روی درصد کمی از اثر حمله تاثیر بگذارد. فرض کنید که $l \leq 2^l$ ، به ازای همه I_i[I]ها $(L \leq i \leq m)$ حمله ما هنوز می تواند با تلاش ۹۵ درصد[۳] موفق شود. یک راه چاره طبیعی در برابر حمله عدم همزمان سازی، ایجاد راهکارهای تصحیح خطای سطح بیت در پایگاه داده است. به هر حال، خطاهای بیت بین IDSهای از بین رفته همانند سایر اطلاعات سری، مانند K1، K2، K3، K4 به سهولت با این روش تصحیح نمی شود. چون استفاده از ترکیب عملیات منطقی (مانند $\oplus$ ، $+\bmod 2^m$) خاصیت جبری توابع آنها را از بین می برد. یک تغییر بیت منفرد در n1 یا n2 می تواند منجر به الگوهای خطای بیت مختلف در مقادیر سری بروز شده شود، جایی که یک مکانیزم تصحیح خطای سازگار باید به کار گرفته شود. بنابراین باعث هزینه های اضافی برای محاسبه و ذخیره در پایگاه داده می شود.

۵,۲ ارسال $\tilde{D}$

یکی از حقه هایی که ما در حمله کاملا افشا کننده خود انجام دادیم تلاش در جهت ایجاد همه A'||B'||C های ممکن و مشاهده پاسخ های موجود از تگ (در مرحله ۳) بود. اگر تگ یک پیام معتبر D را ارسال کند، به این معنی است که تلاش موفق است؛ اگر چنین نباشد، تلاش ناموفق است. فرض کنید که تگ همواره یک پیام $\tilde{D}$ ارسال می کند چه قرائت کننده تصدیق شده باشد و چه نشده باشد (اگر قرائت کننده تصدیق شود $\tilde{D} = D$؛ یا اگر قرائت کننده تصدیق نشده باشد $\tilde{D} \in_R \{0,1\}^m$ است). بنابراین مهاجم نمی تواند هیچ سرنخی در تشخیص یک پیام معتبر D یا یک پیام دلخواه به دست آورد. مهاجم باید آن را به یک قرائت کننده قانونی ارسال کند و منتظر پاسخ بماند. چون برای یک تگ ممکن نیست که یک مقدار تصادفی $\{0,1\}^m$ تولید کند، اگر قرائت کننده تصدیق نشود، تگ می تواند $\tilde{D} = (IDS_{tag(i)}^{(n)} + ID_{tag(i)}) \oplus n2$ را تخصیص دهد. تا زمانی که $\tilde{D}$ برای قرائت کننده قابل تشخیص و برای مهاجم غیر قابل تشخیص است، هر مکانیزم (امن)دیگری کار خواهد کرد.

۵,۳ ذخیره وضعیت

ذخیره برخی الحاقیات در پایگاه داده به تنهایی ممکن است کمک کننده نباشد، اما ذخیره برخی اطلاعات وضعیتی در قرائت

کننده و تگ می تواند برای مواجهه با حملات مفید باشد. شاهد این کار این است که حمله ما بر ناتوانی تگ در تشخیص درخواستهای قرائت کننده قانونی یا تلاشهای یک مهاجم هدف گیری می کند. برای مواجهه با حمله، ضروری است که یک تگ برخی اطلاعات وضعیتی ذخیره شده را در اختیار داشته باشد تا برخی تلاشها از برخی نشست های ادامه دار را نمایان کند. برای این کار، ما یک بیت وضعیت اضافی s را اختصاص می دهیم و s=0 قرار می دهیم، اگر پروتکل با موفقیت کامل (یا همزمان) باشد؛ یا s=1 اگر به دلایلی پروتکل ناکامل (یا غیرهمزمان) باشد. بیت وضعیت پروتکل به منظور نشان دادن تکمیل اجرای پروتکل قرار داده می شود. فقط یک پروتکل کامل شده موفق می تواند عملیات بروز رسانی را در سمت قرائت کننده و تگ راه اندازی کند. این امر به این معنی است که یک مهاجم نمی تواند مقادیر بیتهای n1 یا n2 را با یک پروتکل (غیرکامل) با ایجاد چند تلاش (شکست خورده) تشخیص دهد.

بنابراین پروتکل با وضعیت کامل قرائت کننده و تگ هر دو را برای ذخیره دو عدد تصادفی در آخرین اجرای پروتکل (غیرکامل) در حالت عدم همزمانی نیاز دارد (n1,n2). با یک پروتکل داده شده غیرکامل، تگ انتظار یک پیام تکمیل E (مانند $E=(IDS_{tag(i)}^{(n+1)} + ID_{tag(i)}) \oplus n1 \oplus n2$) از قرائت کننده را دارد. قرائت کننده که اکنون بروز شده است، نیاز به جستجوی پایگاه داده برای محاسبه یک IDS اولیه از تگ با مقادیر ذخیره شده (n1,n2) دارد. اگر یک IDS سابق یافت شود، قرائت کننده پیام تکمیل E را ایجاد می کند و آن را به تگ برای تکمیل پروتکل ارسال می کند. اگر چنین نباشد، یک تگ به صورت دائمی به صورت در خطر افتاده در نظر گرفته می شود.

با فقط یک بیت اضافه شده برای ارائه وضعیت پروتکل و دو عدد تصادفی اضافی (2*96=192 بیت) ذخیره شده در EEPROM، پروتکل جدید اندازه حافظه تگ را ۴۰ درصد (193/96*5) افزایش می دهد، در حالی که همه سخت افزار برای واحدهای منطقی الگوریتم یا واحدهای کنترل تقریبا تغییر نمی کند.

در بالا ما چندین اقدام متقابل در مواجهه با حملات مختلف را پیشنهاد کردیم. در حالی که باید برخی روشهای دیگر برای حمله به پروتکل ها وجود داشته باشد، اقدام متقابل کنونی ممکن است نتواند به دلیل حملات کشف شده پس از آن امنیت را تضمین کند. برخی اقدامات متقابل می تواند برای ایجاد امنیت قوی تر برای پروتکل ها با هم ترکیب شود. به هر حال، روشهای جدید برخی محدودیت ها را برای بکارگیری در برخی وضعیت های واقعی دارد. به عنوان مثال، پروتکل با وضعیت کامل برای همه محیط هایی که قرائت کننده های توزیع شده نمی توانند اطلاعات تگ را به صورت موثر و آنلاین بازیابی کنند تا یک تگ را تصدیق کنند، مناسب نیست.

۶ـ نتیجه گیری و اقدامات آتی

در این مقاله، ما دو حمله موثر در برابر دو پروتکل تصدیق متقابل RFID فوق سبک وزن را که اخیرا پیشنهاد شده اند، ارائه نمودیم. شدت حملات طراحی نا ایمن پروتکل ها را نشان می دهد. کار ما نشان می دهد که استفاده از یک عمل منطقی منفرد برای دستیابی به تصدیق متقابل RFID ایمن تحت مدل مهاجم قدرتمند می تواند کاملا خطرناک باشد. امنیت چنین پروتکل هایی باید با تجزیه و تحلیل رمز استادانه اثبات شود. برای مواجهه با این حملات، برخی اقدامات متقابل نیز به منظور مواجهه با حملات مخرب ارائه شده است. با در نظر گرفتن این حملات و اقدامات متقابل، مرحله بعدی ما طراحی پروتکل تصدیق متقابل RFID فوق سبک وزن و بکارگیری آن در تگهای RFID کم هزینه است.

مراجع

[1] L. Batina, S. Seys, D. Singelee, and I. Verbauwhede, "Hierarchical ECC-based RFID authentication protocol," in RFID. Security and Privacy, ed: Springer, pp. 183-201,2012.

[2] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador, and A. Robagorda, "LMAP: A real lightweight mutual authentication protocol for low-cost RFID tags," in Proc. Of 2nd Workshop on RFID Security, p. 06, 2006.

[3] P. Peris-Lopez, J. C. Hernandez-Castro, J.M Estevez-Tapiador, and A. Ribagorda, "EMAP: An efficient mutual-authentication protocol for low-cost RFID tags," in On the Move to Meaningful Internet Systems, pp. 352-361, 2006.

[4] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador, and A. Ribagorda, "M2AP: A minimalist mutual-authentication protocol for low-cost RFID tags, "in Ubiquitous Intelligence and Computing, ed: Springer, pp. 912-923,2006.

[5] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Tapiador, and A. Ribagorda, "Advances in ultralightweight cryptography for low-cost RFID tags: Gossamer protocol," in Information Security Applications, ed: Springer, pp. 56-68, 2009.

[6] N. J. Hopper and M. Blum, "Secure human identification protocols," in Advances in cryptology-ASIACRYPT, ed: Springer. Pp. 52-66, 2001.