دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکدهٔ مهندسی برق و کامپیوتر

روش‌های رسمی در مهندسی نرم‌افزار

تمرین کامپیوتری Spin

هادی صفری
hadi.safari@ut.ac.ir

مهلت تحویل: نیمه‌شب جمعه ۱۳ دی

راه‌اندازی ابزار Spin

در این تمرین مدل‌سازی یک سیستم با زبان PROMELA و درستی‌یابی آن با ابزار Spin را تمرین خواهید کرد. برای بارگیری Spin می‌توانید به راهنمای آن در آدرس `http://spinroot.com/spin/Man/README.html` مراجعه کنید یا از دستورهای `sudo apt-get install spin` (در Ubuntu Linux) یا `brew install spin` (در macOS با HomeBrew) استفاده کنید.

برای راحتی کار می‌توانید از رابط‌های گرافیکی‌ای که روی Spin طراحی شده‌اند مانند iSpin[۱] و jSpin[۲] نیز کمک بگیرید. توجه داشته باشید که در انتها باید کد خود را با قالب `.pml` به عنوان پاسخ بارگذاری کنید.

توصیه می‌شود از ویرایشگرهای معمول و افزونه‌های[۳] آن‌ها برای توسعهٔ کد خود استفاده کنید.

مدل‌سازی و درستی‌یابی صوری یک پروتکل تبادل کلید

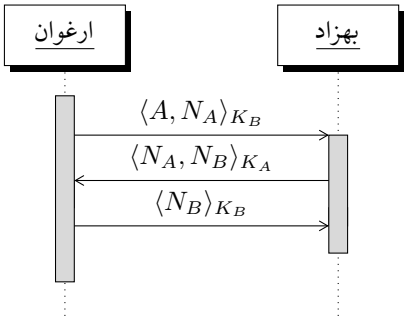
با افزایش اهمیت حفظ امنیت ارتباطات در شبکه‌ها، پروتکل‌های مختلفی برای رمزنگاری پیام‌ها طراحی شده است. این الگوریتم‌ها به دو نوع *متقارن*[۴] و *نامتقارن*[۵] تقسیم می‌شوند. در الگوریتم‌های متقارن رمزنگاری و رمزگشایی پیام‌ها با یک *کلید*[۶] یکتا صورت می‌گیرد. این کلید را نمی‌توان به شکل عمومی منتشر کرد و انتقال آن بین دو طرف ارتباط خودش نیازمند یک کانال ارتباطی امن است. در الگوریتم‌های نامتقارن رمزنگاری با *کلید عمومی*[۷] و رمزگشایی با *کلید خصوصی*[۸] صورت می‌گیرد. این الگوریتم‌ها کندترند ولی به کانال ارتباطی امن برای انتقال کلید نیازی ندارند. یک الگوی رایج برای بهره بردن از مزایای هر دو روش، استفاده از رمزنگاری نامتقارن برای انتقال امن یک کلید و سپس ادامهٔ ارتباط با رمزنگاری متقارن است.

فرض کنید ارغوان می‌خواهد در یک شبکهٔ ناامن با بهزاد صحبت کند. هر کدام از آن‌ها می‌توانند یک عدد یک‌*بارمصرف*[۹]

---

[۱] نیازمند Tcl/Tk Wish
[۲] نیازمند Java
[۳] مانند Sublime-Promela-Spin، language-promela for Atom، code-spin for VSCode و Promela for VS Code

[4] symmetric
[5] asymmetric
[6] key
[7] public key
[8] private/secter key
[9] nonce: **n**umber used **once**



شکل ۱: نحوهٔ کار الگوریتم تبادل کلید

تصادفی تولید کنند که فقط برای همین جلسهٔ ارتباطی معتبر است و با استفاده از رمزنگاری نامتقارن این عدد را به طرف دیگر اعلام کنند. در انتها، هر یک از دو طرف می‌توانند با استفاده از این اعداد یک‌بارمصرف یک کلید مشترک بسازند و ارتباط را با رمزنگاری متقارن ادامه دهند.

هر پروتکل برای تبادل کلید باید در صورت اجرای موفق درستی دو گزاره را تضمین کند:

اصالت [10] هر یک از طرف‌های ارتباط واقعاً همان کسی است که ادعا می‌کند.

محرمانگی [11] فقط دو طرف ارتباط کلید (در این روش اعداد یک‌بارمصرفِ سازندهٔ کلید) را می‌دانند.

۱ مدل‌سازی الگوریتم تبادل کلید بدون حضور مزاحم

ارغوان و بهزاد می‌توانند از سه پیام برای انتقال اعداد یک‌بارمصرفشان استفاده کنند: (شکل ۱)

۱. ارغوان عدد یک‌بارمصرف تصادفی خود را می‌سازد، هویت خود (A) و عدد یک‌بارمصرفش (N_A) را با کلید عمومی بهزاد (K_B) رمزنگاری می‌کند و برای بهزاد می‌فرستد. ($\langle A, N_A \rangle_{K_B}$)

۲. بهزاد پس از این که ارغوان می‌خواهد مکالمه‌ای را شروع کند، عدد یک‌بارمصرف تصادفی می‌سازد، عدد یک‌بارمصرف ارغوان (N_A) و عدد یک‌بارمصرف خودش (N_B) را با کلید عمومی K_A رمزنگاری می‌کند و برای ارغوان می‌فرستد. ($\langle N_A, N_B \rangle_{K_A}$)

۳. ارغوان پس از دریافت پیام بهزاد، عدد یک‌بارمصرف بهزاد (N_B) را با کلید عمومی بهزاد (K_B) رمزنگاری می‌کند و برای بهزاد می‌فرستد تا نشان دهد به کلید خصوصی ارغوان دسترسی دارد و واقعاً ارغوان است. ($\langle N_B \rangle_{K_B}$)

سیستمی را شامل ارغوان و بهزاد و یک ارتباط کامل از ارغوان به بهزاد با زبان PROMELA مدل‌سازی کنید.

مدل‌سازی یک ارتباط از ارغوان به بهزاد کافی است؛ بنابراین می‌توانید فرض کنید اعداد یک‌بارمصرف هر یک از آن‌ها یک عدد مشخص است. برای سادگی مدل، می‌توانید فرض کنید همهٔ ارتباطات به شکل قرار ملاقات[12] صورت می‌گیرند. ممکن است بخواهید در مدل‌سازیتان پیام سوم را به شکل $\langle N_B, \emptyset \rangle_{K_B}$ تصور کنید تا پیام‌ها یک‌اندازه شوند. برای مدل‌سازی رمزنگاری نامتقارن، می‌توانید یک پارامتر برای مشخص کردن هویت کسی که پیام با کلید عمومی او رمزنگاری شده به پیام اضافه کنید. هنگام خواندن پیام، هر کس فقط وقتی باید بتواند محتویات رمزنگاری‌شدهٔ پیام را بخواند که با کلید عمومی خودش رمزنگاری شده باشد.

آیا سیستم دچار بن‌بست[13] می‌شود؟

آیا ویژگی‌های اصالت و محرمانگی در این سیستم حفظ می‌شوند؟ برای هر یک از آن‌ها یک ویژگی LTL تعریف کنید و برقرار بودن هر یک از آن‌ها را بررسی کنید.

[10] authenticity
[11] confidentiality
[12] rendezvous
[13] deadlock

۲ افزودن مزاحم منفعل

یک مزاحم[۱۴] به مدل خود اضافه کنید. این مزاحم به تمام ارتباطات شبکه دسترسی دارد و می‌تواند به جای ارغوان یا بهزاد پیام‌هایشان را بردارد. هر کس ــ از جمله مزاحم ــ فقط وقتی می‌تواند محتویات رمزنگاری‌شدهٔ پیام‌ها را بخواند که با کلید عمومی خودش رمزنگاری شده باشد. مزاحم حافظه‌ای به اندازهٔ یک پیام دارد و می‌تواند آخرین پیامی را که برداشته برای هر کسی که بخواهد دوباره ارسال کند.

نیازی به بررسی بن‌بست نداشتن[۱۵] سیستم نیست.

آیا اصالت و محرمانگی در سیستم حفظ می‌شود؟

۳ تبدیل مزاحم به مزاحم فعال

فرض کنید ارغوان و بهزاد مزاحمی را که در مرحلهٔ قبل به سیستم افزودید به عنوان یک شخص ثالث به رسمیت می‌شناسند. بنابراین باید قابلیت‌هایی را به مدلتان اضافه کنید. ارغوان ممکن است برای بهزاد یا مزاحم پیام بفرستد. بهزاد ممکن است از ارغوان یا مزاحم پیام دریافت کند. مزاحم ممکن است پیام‌های افراد دیگر به هر کسی را بردارد و ذخیره کند، پیام‌هایی را که ذخیره کرده عیناً برای هرکسی دوباره بفرستد، و پیام جدیدی برای ارغوان یا بهزاد بفرست.

ممکن است این پیام‌ها محتویات نادرستی داشته باشند؛ مثلاً N_A در یک پیام نوع سوم برای خود ارغوان ارسال شود. از آن‌جا که ارغوان و بهزاد مزاحم را به عنوان یک شخص ثالث به رسمیت می‌شناسند، ممکن است پیامی را با کلید عمومی او رمزنگاری کنند و برایش ارسال کند. در این صورت مزاحم می‌تواند محتویات داخلی آن پیام را بخواند و اگر عدد یک‌بارمصرفی در آن بود آن عدد را ذخیره کند.

پیاده‌سازی مدل برای یک ارتباط ارغوان و بهزاد کافی است؛ بنابراین همچنان می‌توانید فرض کنید عدد یک‌بارمصرف هر یک از آن‌ها یک مقدار مشخص است. مهاجم می‌تواند به تعداد دلخواه هر یک از کارهایی را که گفته شد بکند؛ اما برای سادگی، فرض کنید مهاجم همواره یک مقدار مشخص را به عنوان عدد یک‌بارمصرف خودش به کار می‌گیرد.

نیازی به بررسی بن‌بست‌نداشتن سیستم نیست.

با استفاده از ابزار وارسی مدل Spin، یک سناریوی ناقض اصالت یا محرمانگی در این سیستم بیابید. آیا روشی برای وصله زدن[۱۶] الگوریتم به ذهنتان می‌رسد تا این سناریو اتفاق نیفتد؟

نحوهٔ تحویل

لطفاً فقط کد نهایی خود و یک سناریوی ناقض اصالت یا محرمانگی را در پرونده‌ای به نام `SID.pml` یا `SID.zip` (که SID شمارهٔ دانشجویی شماست) پیش از اتمام مهلت تحویل در صفحهٔ وب درس بارگذاری کنید. نیازی به تحویل کدهای مراحل میانی یا وصله زدن الگوریتم نیست؛ مراحل میانی صرفاً برای راهنمایی شما در انجام پروژه بیان شده‌اند. سناریوی ناقض اصالت یا محرمانگی را می‌توانید به شکل توضیح[۱۷] در انتهای کد یا در یک پروندهٔ جدا (با قالب `.pdf`) بنویسید.

توجه داشته باشید که این تمرین تحویل حضوری دارد و در صورت حضور نیافتن در تحویل حضوری نمره‌ای از این تمرین دریافت نخواهید کرد.

منابع و مراجع

- http://www.cse.chalmers.se/edu/year/2019/course/TDA294_Formal_Methods_for_Software_Development/Lab1.html
- ftp://ftp.lab.unb.br/pub/netlib/spin/ws02/maggi.pdf

[14] intruder
[15] deadlock-freeness
[16] patch
[17] comment