



## تمرین کامپیوتری شماره ۱ مبانی امنیت شبکه، بهار ۱۴۰۰

در این تمرین قصد داریم یک نسخه شخصی از RSA را پیاده سازی کنیم. به پیاده سازی RSA مطابق آنچه در منابع آموزشی به آن پرداخته می شود، Textbook RSA می گویند. کد تحویلی شما می بایست شامل ۳ فایل مجزا به شرح زیر باشد:

### (۱) فایل keygen.py

- در این قسمت شما می بایست کلیدهای خصوصی و عمومی را تولید نمایید. برنامه شما در ابتدای اجرا، دو عدد اول  $P$  و  $Q$  را از کاربر تقاضا می کند. برنامه باید اول بودن هر دوی این اعداد را بررسی کرده و برابر نبودن آن ها را احراز کند. در صورت عدم برقراری شرایط، با پیغام خطای مناسب اجرای برنامه متوقف می شود. همچنین با توجه به نوع رمزنگاری (که در قسمت های بعد شرح داده خواهد شد)، مقدار پیمانه  $n$  (برابر حاصلضرب  $P$  و  $Q$ ) بایستی بزرگتر مساوی ۲۵۶ بوده و در غیر این صورت از اجرای برنامه ممانعت به عمل آید.
- پس از این که مقدار  $P$  و  $Q$  مناسب توسط برنامه دریافت شد، برنامه به صورت خودکار یک عدد تصادفی  $e$  مطابق با شرایط الگوریتم RSA تولید می کند. در این مرحله، توجه به جزئیات شروط لازم برای  $e$  مورد نیاز است. برنامه شما باید در تمامی فضای حالت به صورت تصادفی جستجو کرده و یک  $e$  مطلوب و تصادفی را بیابد.
- پس از یافتن یک  $e$  مطلوب، برنامه شما باید به تولید  $d$  مبادرت ورزد. یافتن  $d$  باید صرفاً توسط الگوریتم اقلیدسی انجام گیرد و پیاده سازی آن ذیل یک تابع الزامی است. در این مرحله برای پیدا کردن  $d$  از brute force استفاده نکنید.
- در انتها، برنامه شما باید به ترتیب مقادیر  $P, Q, n, \Phi(n), e$  و  $d$  را در خروجی کنسول چاپ نماید.
- کلید خصوصی شامل زوج مقادیر  $d$  و  $n$  می باشد. برنامه شما در انتهای اجرا می بایست یک فایل با نام `private.key` تولید کرده که در خط اول آن مقدار  $d$  و در خط دوم، مقدار  $n$  چاپ شده باشد.
- به همین ترتیب، کلید عمومی نیز در یک فایل با نام `public.key` که خط اول آن شامل مقدار  $e$  بوده و خط دوم شامل  $n$  است باید درج شود.

### (۲) فایل encrypt.py

- این فایل به انجام رمزنگاری روی فایل ورودی با نام `input.txt` پرداخته و با استفاده از کلید عمومی موجود در فایل `public.key` عملیات رمزنگاری را انجام می دهد. فایل ورودی می بایست به صورت باینری (و نه متنی) خوانده شود.
- با توجه به محدودیت طول پیمانه در الگوریتم RSA و طول زیاد فایل های ورودی، نیاز است تا رمزنگاری در این قسمت به صورت بلوکی انجام گیرد. طول هر بلوک رمزنگاری ۱ بایت (معادل ۸ بیت) بوده و عملیات رمز به ازای هر یک بایت صورت خواهد گرفت.
- طبق مباحث الگوریتم رمزنگاری RSA، اگر طول پیغام  $M$  بوده و طول پیمانه نیز  $n$  باشد ( $n > M$ )، طول  $C$  لزوماً به اندازه  $M$  نیست؛ ولی از  $n$  کمتر خواهد بود. این خصوصیت الگوریتم RSA باعث می شود که عملیات بلوکه سازی فایل ورودی به راحتی رمزنگارهای بلوکی مانند DES و یا AES نباشد، زیرا طول خروجی یک بلوک رمز RSA برابر ورودی آن نبوده و حتی بلوک های متوالی، خروجی هایی با طول های یکسان نیز ایجاد نخواهند کرد. برای حل این مشکل، تمام



## تمرین کامپیوتری شماره ۱ مبانی امنیت شبکه، بهار ۱۴۰۰

بلوک‌های خروجی فرآیند رمزنگاری فایل را با طول یکسان و برابر ماکسیمم طول ممکن در ایجاد یک بلوک C در نظر می‌گیریم.

- یک پیغام ۸ بیتی M ماکسیمم مقدار ۲۵۵ را می‌تواند اتخاذ کند. فلذا با توجه به شرط  $n > M$ ، مقدار n می‌بایست حداقل برابر ۲۵۶ باشد تا در فرآیند رمزنگاری هر بلوک هشت بیتی با مقدار دلخواه، مشکلی ایجاد نشود. به این ترتیب، در صورتی که n بزرگتر مساوی ۲۵۶ باشد، مقدار C نیز می‌تواند مقداری از عدد ۰ تا  $n-1$  باشد. این بدین معنیست که ماکسیمم طول C برابر تعداد بیت‌های لازم برای نگهداری عدد  $n-1$  می‌باشد.
- با توجه به اینکه واحدهای تشکیل دهنده یک فایل کمتر از ۱ بایت نمی‌توانند باشند، مقادیر ذکر شده می‌بایست به نزدیک ترین ضریب بایت (هشت تایی) کران بالا گرد شود. با توجه به این نکته، در صورتی که n برابر ۲۵۶ باشد، ماکسیمم ظرفیت مورد نیاز برای هر بلوک C برابر ۱ بایت؛ در صورتی که n بزرگتر از ۲۵۶ و کوچکتر مساوی ۶۵،۵۳۶ باشد ماکسیمم سایز بلوک C برابر ۲ بایت، بزرگتر از ۶۵،۵۳۶ و کوچکتر مساوی ۱۶،۷۷۷،۲۱۶ باشد ماکسیمم ۳ بایت و به همین ترتیب.
- برنامه رمزنگار می‌بایست با بررسی مقدار n موجود در کلید عمومی، مقدار بایت مورد نیاز برای بلوک‌های C را بدست بیاورد و این مقدار را به صورت یک عدد (بدون توضیحات اضافه) در خروجی برنامه (stdout) چاپ کند.
- پس از این، هر بلوک ۱ بایتی از فایل ورودی را رمز کرده و حاصل رمز را (که می‌تواند ۱ بایت، ۲ بایت، ۳ بایت و یا بیشتر بسته به مقدار n باشد) در فایل خروجی به نام input.txt.enc ذخیره کند. در این صورت، فایل input.enc سایز یک برابر، دو برابر، سه برابر و یا بیشتر (بسته به مقدار n) نسبت به فایل اصلی پیدا خواهد کرد.

### ۳) فایل decrypt.py

- این فایل با استفاده از کلید خصوصی موجود در فایل private.key به رمزگشایی یک فایل ورودی با نام input.txt.enc می‌پردازد. این برنامه در ابتدا محتوای کلید خصوصی را قرائت کرده و بسته به مقدار n، تعداد بایت‌هایی که می‌بایست به عنوان یک بلوک C در نظر بگیرد را بدست می‌آورد و این مقدار را به صورت یک عدد (بدون توضیحات اضافه) در خروجی برنامه (stdout) چاپ کند. این مقدار می‌تواند برابر ۱ بایت، ۲ بایت، ۳ بایت و یا بیشتر بسته به مقدار n خواهد بود.
- پس از این، در هر نوبت به خواندن C با سایز مورد نظر از فایل ورودی کرده و با انجام عملیات رمزگشایی RSA روی آن، M را که یک مقدار ۱ بایتی بوده بدست آورده و آن را در فایلی با نام output.txt ذخیره می‌نماید. فایل output.txt می‌بایست کاملاً مشابه input.txt بدست بیاید.



## تمرین کامپیوتری شماره ۱ مبانی امنیت شبکه، بهار ۱۴۰۰

### نکات

\* برنامه شما باید علاوه بر رمزنگاری و رمزگشایی فایل‌های متنی، توانایی انجام عملیات روی هر نوع فایلی را داشته باشد. این امر با برآورد شرط مشابهت صد درصدی بابت به بایت فایل‌های `input.txt` و `output.txt` تضمین خواهد شد؛ فلذا با تغییر نام هر گونه فایل متنی، مالتی‌مدیا و غیره به نام‌های مربوطه، انجام عملیات رمزنگاری و بازگرداندن پسوند آن‌ها پس از رمزگشایی، فایل‌ها بایستی مشابهت کامل و هویت خود را حفظ نمایند. برای این کار لازم است تا خواندن و نوشتن مقادیر فایل‌های ورودی و خروجی را به صورت باینری (و نه متنی) انجام دهید.

\* برنامه‌های نوشته شده می‌بایست Robustness لازم را دارا باشد و در هر مرحله با پیغام خطای مناسب علت عدم اجرای صحیح را اطلاع رسانی کند. تحت هیچ شرایطی بروز Segmentation Fault و یا Exception ها و Error های پیش فرض در روند اجرای برنامه قابل قبول نیست.

\* تمامی الگوریتم‌های لازم می‌بایست به صورت کامل پیاده‌سازی شوند. تنها استفاده از توابع ریاضی توان، لگاریتم و اپراتورهای ریاضی پیش‌فرض قابل قبول است.

\* متغیرهای برنامه شما می‌بایست تحمل اجرای برنامه‌ای با پیمانه  $n$  کوچکتر مساوی ۴,۲۹۴,۹۶۷,۲۹۶ را داشته باشد (ماکسیمم ۴ بایت).

\* برنامه‌های شما می‌بایست با زبان Python نوشته شده باشد.

\* استفاده از هرگونه کد قبلی و نمونه کدهای آنلاین غیر مجاز است. تمامی مراحل و توابع می‌بایست توسط خود شما پیاده‌سازی گردد.

\* در این تمرین، تأکیدی روی Performance نیست. تمرکز خود را روی پیاده‌سازی دقیق الگوریتم قرار دهید.

\* ملاک جزئیات الگوریتم‌ها، مطالب تدریس شده در کلاس درس می‌باشد.

\* با توجه به تصحیح خودکار تمرین‌ها، لطفاً توجه لازم را به نام‌گذاری فایل‌ها، ورودی‌ها و خروجی‌ها مبذول دارید.

\* به شماتیک ضمیمه این تمرین توجه لازم را داشته باشید.

\* فایل‌های تحویلی شما می‌بایست یک فایل فشرده با فرمت zip با نام شماره دانشجویی شما و فقط شامل ۳ فایل سورس `keygen.py`، `encrypt.py` و `decrypt.py` باشد.