به نام خداوند بخشنده‌ی مهربان

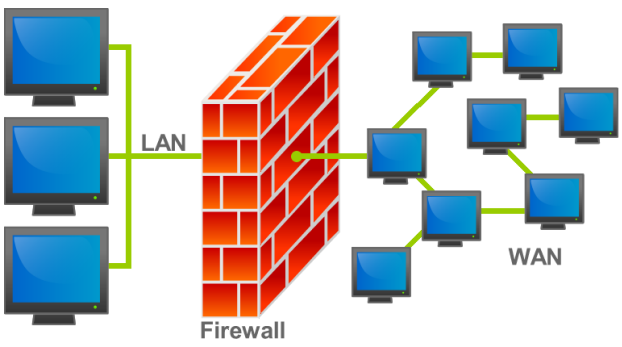
تمرین برنامه‌نویسی یکم[1]
شبکه‌های کامپیوتری

مدرس: مهدی خرازی **بهار ۱۳۹۱**

در این تمرین شما ضمن **آشنایی با سیستم پرتو**، یک دیواره‌ی آتش ساده خواهید نوشت.

مقدمه

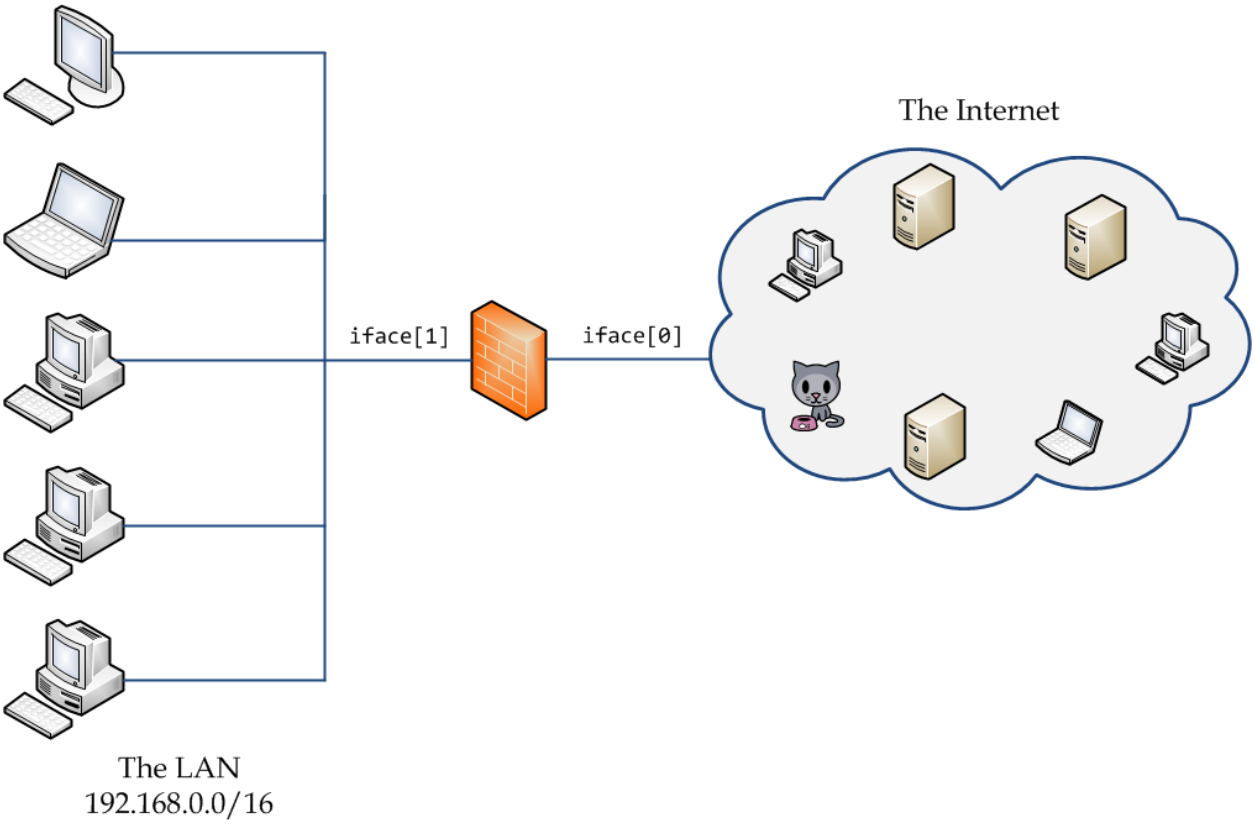
«دیواره‌ی آتش» یا Firewall برنامه‌ای است که از دسترسی غیرمجاز به منابع شبکه جلوگیری می‌کند. برای نمونه فرض کنید که یک شرکت دارای یک شبکه‌ی محلی است و این شبکه‌ی محلی از طریق یک Gateway به اینترنت متصل است. ممکن است یک مهاجم بخواهد از طریق اینترنت به این شبکه‌ی محلی دسترسی پیدا کند و اقدام به خرابکاری نماید. برای جلوگیری از چنین اتفاقی، یک راه قطع کردن شبکه‌ی محلی از اینترنت است؛ اما راه منطقی در این میان استفاده از یک Middleware بین شبکه‌ی محلی و اینترنت خواهد بود که این ارتباط را کنترل نماید. این سیستم را به‌گونه‌ای تنظیم می‌کنیم که اجازه‌ی ورود هیچ بسته‌ای از بیرون به درون شبکه‌ی محلی وجود نداشته باشد و در عوض شبکه‌ی محلی بتواند به بیرون بسته ارسال کند.



شکل ۱ ــ یک نمونه از قرارگیری دیواره‌ی آتش در یک پیکربندی شبکه

[1] با تشکر از بهنام مومنی، امیر شیخها، امیرمهدی احمدی‌نژاد، کامیار اللهوردی، علی‌محمد ربانی، سجاد فولادی و مهرداد مرادی

در این تمرین قصد داریم یک دیواره‌ی آتش ساده طراحی کنیم. این دیواره‌ی آتش به عنوان یک گره‌ی میانی بین شبکه‌ی محلی و شبکه‌ی اینترنت قرار می‌گیرد. شما وظیفه‌ی نوشتن این گره را بر عهده دارید. شکل زیر ساختار کلی را نشان می‌دهد:



شکل ۲ ــ پیکربندی مورد استفاده در این تمرین

دیواره‌ی آتش دارای دو interface است؛ یک interface (iface[0]) از طریق یک Gateway به شبکه‌ی اینترنت و interface دیگر به شبکه‌ی داخلی وصل است. بسته‌ای که از شبکه‌ی داخلی به شبکه‌ی بیرونی می‌رود outgoing و بسته‌ای را که از بیرون به داخل شبکه می‌آید incoming می‌نامیم.

این دیواره‌ی آتش دارای تعدادی قاعده است که این قواعد دارای ترتیب می‌باشند (در مورد اینکه این قواعد چگونه در اختیارتان قرار می‌گیرد، در ادامه صحبت خواهیم کرد). برای نمونه یک جدول قواعد برای دیواره‌ی آتش می‌تواند به صورت زیر باشد:

جدول ۱ ــ یک نمونه از جدول قواعد برای یک دیواره آتش

#	src-ip	dst-ip	protocol	action
1	*	*	17	ALLOW
2	*	97.107.141.111	6	ALLOW
3	192.168.0.0/16	*	6	DENY
4	*	*	*	ALLOW

هر بسته‌ای که دریافت می‌شود به ترتیب با هر کدام از قواعد این جدول کنترل می‌شود. در صورتی که بسته مطابق قاعده باشد، عملیات بخش action روی آن صورت می‌گیرد؛ یا فرمان ALLOW است، یعنی فایروال به آن اجازه‌ی عبور می‌دهد (در واقع بسته را بعد از بررسی باید روی interface متناسب با آن ارسال نماید)، یا فرمان DENY است که در این صورت باید بسته drop شود. در صورتی که بسته مطابق قاعده نبود، قاعده‌ی بعدی چک می‌شود.

در صورتی که در انتها قاعده‌ای مطابق بسته یافت نشود، بسته را drop می‌کنیم[۲].

[۲] به طور کلی دو دسته سیاست امنیتی می توان برای این شرایط در نظر گرفت؛ سیاست باز که در آن به همه‌ی بسته‌ها اجازه‌ی عبور داده می‌شود، مگر آنکه خلاف آن بیان شود و سیاست بسته که در آن فرض بر رد کردن تمام بسته‌ها به صورت پیش‌فرض است.

محیط

جهت آشنایی با سیستم پرتو لازم است که برنامه خود را بر روی «چارچوب کاربر» این سیستم پیاده کنید. پرتو این امکان را به شما می‌دهد که تعدادی گره مجازی در شبکه داشته باشید و آن‌ها را مطابق میل خود برنامه‌ریزی کنید. برای آشنایی بیشتر توصیه می‌شود که به مستند «راهنمای چارچوب کاربر» مراجعه فرمایید.

انتظارات

شما در این تمرین باید برنامه‌ی firewall را بنویسید. به عنوان ورودی به این برنامه یک فایل حاوی قواعدی که باید firewall بر اساس آن‌ها عمل کند داده می‌شود.

توپولوژی

توپولوژی اولیه که در اختیار شما قرار می‌گیرد مانند شکل ۱ است. به نحوه‌ی نامگذاری‌ها دقت کنید. همیشه iface[0] متصل به شبکه‌ی بیرونی و iface[1] متصل به شبکه‌ی داخلی است. محدوده‌ی IPهای شبکه‌ی داخلی با نمادگذاری[3] CIDR در Custom Information در اختیار شما قرار خواهد گرفت.

جزئیات برنامه

فایل قواعد

به عنوان یک آرگومان ورودی برنامه، آدرس یک فایل حاوی قواعدی که باید در firewall لحاظ شوند داده می‌شود. در هر خط از این فایل یک قاعده آمده است. هر قاعده دارای ساختار زیر است:

<action> <key-value pairs> *[protocol-specific rules]*

جدول زیر هر کدام از این بخش‌ها را بیشتر توضیح می‌دهد:

جدول ۲ ـ اجزای اصلی یک قاعده

action	اینکه باید بسته‌ی مطابق با این قاعده قبول شود یا نه. می‌تواند دارای دو مقدار ALLOW و DENY باشد.
key-value pairs	این بخش شامل چند جفت مقدار است. این مقادیر ویژگی‌های بسته را مشخص می‌کنند. هر کلید با فاصله از مقدارش جدا می‌شود. مقادیری که می‌توانند به عنوان کلید بیایند در جدول بعدی آمده‌اند.
protocol-specific rules	اگر پروتکل بسته‌ی ارسالی یکی از پروتکل‌های TCP یا UDP باشد، می‌توان تعدادی قاعده متناسب با هرکدام از این پروتکل‌ها در این قسمت نوشت. این بخش تنها در صورتی می‌تواند در قاعده قرار گیرد که

[3] notation

	مقدار protocol تنظیم شده باشد.

مقادیر کلید می‌توانند از جدول زیر انتخاب شوند:

جدول ۳ ـ کلیدهای اصلی برای قاعده

src-ip	آدرس IP فرستنده‌ی بسته را مشخص می‌کند. مثلاً 192.168.0.10. همچنین این مقدار می‌تواند با استفاده از نمادگذاری CIDR یک محدوده‌ی آی‌پی را هم مشخص کند؛ مثلاً 192.168.112.0/8.
dst-ip	آدرس IP گیرنده‌ی بسته را مشخص می‌کند. همانند بالا می‌تواند یک محدود را با نمادگذاری CIDR مشخص کند.
protocol	پروتکل بسته را مشخص می‌کند. مثلاً مقدار 17 برای بسته‌ی UDP و 6 برای بسته‌ی TCP.

برای نمونه، برای اینکه بسته‌ای که از IP مبدأ 97.107.141.111 می‌آید و پروتکل آن UDP است فیلتر شود، می‌توان قاعده‌ی زیر را نوشت:

DENY --src-ip 97.107.141.111 --protocol 17

در صورتی که پروتکل TCP باشد، در قسمت protocol-specific rules می‌توان از کلیدهای زیر استفاده کرد:

جدول ۴ ـ کلیدهای مربوط به پروتکل TCP

src-port	پورت فرستند بسته
dst-port	پورت گیرنده بسته
tcp-flags	تعیین می‌کند که کدام flagها برای بسته‌ی TCP باید روشن باشند. این فیلد می‌تواند دارای شش مقدار باشد: URG، ACK، PSH، RST، SYN و FIN. در صورتی که بخواهیم چند flag را به طور همزمان برای بسته شرط کنیم، آنها را با & از هم جدا می‌کنیم. مثلاً SYN&ACK؛ یعنی بسته باید دارای هم پرچم SYN باشد و هم پرچم ACK. دقت کنید که فاصله‌ای بین & و پرچم‌ها نیست.

در صورتی که پروتکل UDP باشد هم کلیدهای زیر قابل استفاده هستند:

جدول ۵ ـ کلیدهای مربوط به پروتکل UDP

src-port	پورت فرستند بسته
dst-port	پورت گیرنده بسته

چند نکته:

۱. برای اینکه یک بسته با یک قاعده مطابق شود، باید تمام شرایط ذکر شده در قاعده با آن مطابق باشد؛ در واقع شرط‌ها با یکدیگر AND منطقی می‌شوند.

۲. کلیدهایی که در قاعده ذکر نمی‌شوند، "Don't Care" تلقی می‌شوند.

۳. قبل از پردازش بسته‌ها، باید مقادیر Checksum آن‌ها کنترل شود و در صورتی که این مقادیر نادرست باشند، بسته Drop می‌شود.

برای نمونه، برای مسدود کردن تمام بسته‌ی TCP که آدرس IP مقصد آن‌ها 213.233.168.15 است و به پورت 80 ارسال می‌شوند و پرچم SYN آنها روشن است، می‌توان قاعده‌ی زیر را نوشت:

DENY --dst-ip 213.233.168.15 --protocol 6 --dst-port 80 --tcp-flags SYN

برنامه‌ی Firewall

این برنامه در خط فرمان به صورت زیر اجرا می‌شود:

./firewall.sh <rules_file>

rules_file نام فایلی است که قواعد Firewall در آن قرار دارند.

Custom Information مربوط به این Firewall شامل موارد زیر است[۴]:

۱. خط ۱، محدوده‌ی IPهای شبکه‌ی داخلی را با نمادگذاری CIDR نشان می‌دهد. برای مثال: 192.168.0.0/24.

۲. خط ۲، آدرس IP مربوط به Gateway اینترنت را نشان می‌دهد. بسته‌هایی که باید به شبکه‌ی بیرونی بروند، به این Gateway ارسال می‌شوند.

برای مثال یک نمونه Custom Information به صورت زیر است:

```
192.168.0.0/24
213.233.168.1
```

پروتکل ARP

برای ارسال بسته‌ها، لازم است که آدرس MAC برای گره مقصد را در اختیار داشته باشید (در سرآیند Ethernet باید آدرس MAC گره‌ی «بعدی» قرار گیرد؛ این درحالی است که در سرآیند IP آدرس فرستند و گیرنده قرار دارد). در صورتی که مقصد بسته شبکه‌ی بیرونی است، این آدرس MAC، آدرس Gateway خواهد بود. برای اینکه بتوانیم از روی IP مقصد، آدرس MAC

[۴] برای دریافت کردن Custom Information مربوط به هر ماشین از تابع getCustomInformation() استفاده کنید.

مقصد را بدست آوریم، از پروتکل ARP استفاده می‌کنیم. در این پروتکل با داشتن یک آدرس IP می‌توانیم آدرس MAC هدف را بدست آوریم.

در این تمرین، برای بدست آوردن آدرس‌های MAC باید پروتکل ARP را پیاده‌سازی کنید. می‌توانید برای سادگی یک جدول از زوج‌های IP/MAC نگه‌داری نمایید و برای هر آدرس IP تنها یک‌بار از پروتکل ARP استفاده کنید.

پروتکل ARP از یک سیستم پیام ساده استفاده می‌کند که هر پیام شامل درخواست یا پاسخ به یک درخواست است. اندازه‌ی بسته‌ی ARP وابسته به اندازه‌ی آدرس‌های لایه‌ی پایین‌تر (مثلاً Ethernet) و لایه‌ی بالاتر (مثلاً IPv4) است. سرآیند ARP نوع این لایه‌ها و اندازه‌ی آدرسها را در خود مشخص می‌کند. همچنین یک کد Operation در قسمت سرآیند قرار دارد که مشخص می‌کند بسته درخواست است یا پاسخ به درخواست.

فرمت کلی بسته‌های ARP در جدول زیر آمده است:

بیت‌ها	نام	توضیحات
0 – 15	HTYPE	این فیلد نوع پروتکل لایه‌ی پایین‌تر را مشخص می‌کند. مثلاً برای Ethernet این مقدار برابر با 1 است.
16 – 31	PTYPE	نوع پروتکل لایه‌ی بالاتر توسط این بخش مشخص می‌شود. برای IPv4 این مقدار برابر 0x8000 خواهد بود.
32 – 39	HLEN	طول آدرس لایه‌ی پایین‌تر (آدرس سخت‌افزار) را به بایت مشخص می‌کند. مثلاً برای Ethernet این مقدار برابر با 6 است.
40 – 47	PLEN	طول آدرس لایه‌ی بالاتر (که نوع آن را PTYPE معین کرده‌است). مثلاً برای IPv4 این مقدار 4 است.
48 – 63	OPER	نوع عملیات؛ مقدار ۱ برای درخوسات و ۲ برای پاسخ آن.
64 – 111	SHA	آدرس سخت‌افزاری (مثلاً آدرس MAC برای Ethernet) فرستنده.
112 – 143	SPA	آدرس لایه‌ی بالاتر فرستنده.
144 – 191	THA	آدرس سخت‌افزاری گیرنده. برای درخواست مقدار این فیلد اهمیتی ندارد.
192 – 224	TPA	آدرس لایه‌ی بالاتر فرستند. برای درخواست، مقدار این فیلد را برابر آدرسی قرار می‌دهیم که می‌خواهیم MAC آن را به دست آوریم.

دقت کنید که بسته‌ی درخواست باید در شبکه Broadcast شود؛ به این منظور باید آدرس MAC مقصد برابر با ff:ff:ff:ff:ff:ff قرار داده شود.

دقت کنید که پروتکل ARP در گره‌ی Firewall غیرفعال است و شما باید بسته‌های ورودی ARP (یعنی Requestهای ARPی که ممکن است به این گره ارسال شود) را بررسی کنید و در صورتی که بسته متعلق به شما بود، آن را پاسخ دهید (برای مثال Gateway ممکن برای بدست آوردن آدرس MAC مربوط به Firewall درخواست ARP ارسال نماید که باید این بسته توسط شما پاسخ داده شود).

جزئیات بیشتر پروتکل ARP و ساختار بسته‌های آن را می‌توانید از آدرس زیر بدست آورید:

http://en.wikipedia.org/wiki/Address_Resolution_Protocol

نکات ضروری

- برای آشنایی با سرایند لایه‌های به کار رفته در این تمرین، به لینک‌های زیر مراجعه کنید:
  [1] http://www.security-freak.net/raw-sockets/raw-sockets.html
  [2] http://en.wikipedia.org/wiki/IPv4
  [3] http://tools.ietf.org/html/rfc1035
  [4] http://en.wikipedia.org/wiki/Address_Resolution_Protocol
- در صورتیکه هر مشکل یا پرسشی داشتید که فکر می‌کنید پاسخ آن برای همه مفید خواهد بود، لطفاً آن را به گروه پستی درس ارسال کنید.
- از فرستادن جواب تمرین به گروه پستی جداً خودداری کنید.
- فرستادن کل یا قسمتی از برنامه‌تان برای افراد دیگر، یا استفاده از کل یا قسمتی از برنامهٔ فردی دیگر به نام خود، تقلب محسوب می‌شود.
- پس از اتمام کارتان لازم است که پوشه user را به همراه Makefile فشرده کرده و بر روی سیستم خودکار داوری[5] upload کنید.

موفق باشید

[5] http://partov.sharif.edu/