

Скрытые Каналы. Лабораторная работа #3.

Соколов А.Д. Б20-505

Вариант 7

Модель 4. Пакеты случайной длины передаются в случайные моменты времени.

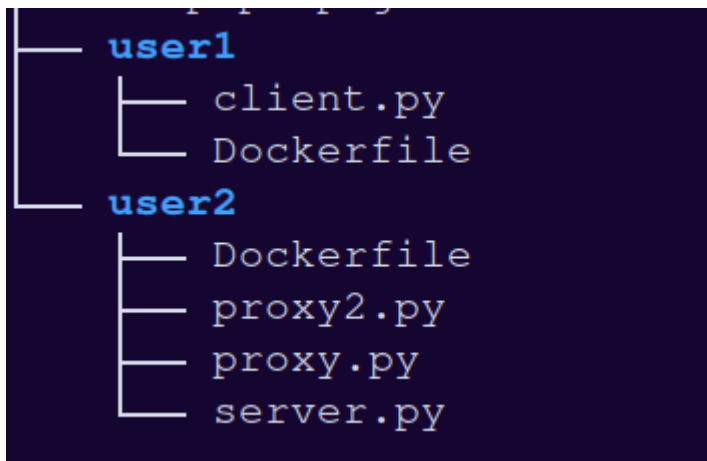
Пример 9. Time Relay Covert Channel

Исходя из измерения возможных значений задержек между пакетами в открытом трафике, определяется медианное значение этого множества, и при передаче скрытого сообщения для кодирования 0 выбирается значение межпакетного интервала из подмножества значений, которые больше медианного, а для 1 — которые меньше

Возможности закладки:

- Буферизация трафика
- Генерация фиктивного трафика

Архитектура



Есть сокет сервера, который непрерывно слушает.

Есть сокет закладки, которая непрерывно слушает и делает свои дела с полученными пакетами.

Есть сокет защиты, который непрерывно слушает и защищает

Есть клиент - сокет который отправляет n пакетов на порт закладки.

Реализация

client.py

```

1  import argparse
2  import socket
3  from os import urandom
4  from random import randint, random
5  from time import sleep
6
7
8  ✓ def model_packets(host, port, n_packets=200):
9      client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10     client.connect((host, port))
11
12     sent = 0
13     while sent < n_packets:
14         data = urandom(randint(1, 100))
15         data = len(data).to_bytes(4, "big") + data
16         client.send(data)
17
18         sent += 1
19         sleep(random())
20     client.close()
21
22
23  if __name__ == "__main__":
24     parser = argparse.ArgumentParser()
25     parser.add_argument(
26         "-p", "--port", help="server running port", type=int, required=True
27     )
28     parser.add_argument("--host", help="server running host", type=str, required=True)
29     args = parser.parse_args()
30
31     model_packets(args.host, args.port)

```

proxy.py

```
1  import socket
2
3  from logging import error
4  import argparse
5
6  from time import sleep
7  from random import random, randint
8  from os import urandom
9
10 from queue import Queue
11 from threading import Thread
12
13 from termcolor import colored
14
15 # from concurrent.futures import ThreadPoolExecutor
16
17 M = 0.48871803283691406
18
19
20 ✓ def start_proxy(port: int, server_host: str, server_port: int, message: str):
21     q = Queue()
22     msg = int.from_bytes(message.encode(), "big")
23     finished = False
24
25     ✓ def listen_and_buffer(port):
26         sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
27         sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
28         sock.bind(("localhost", port))
29         sock.listen(1)
30
31         client, addr = sock.accept()
32         error(colored(f"PROXY::Connection from {addr}", "red"))
33
34         while True:
35             data = client.recv(1024)
36             error(colored(f"PROXY::Received data {data}", "red"))
37             if not data:
38                 break
39             q.put(data)
40         client.close()
41
42     global finished
43     finished = True
44
```

```

45 ✓ def be_proxy_and_send_message(host: str, port: port, msg):
46     data = b""
47     delay = 0.0
48     started = False
49     server = None
50     while msg or not finished:
51         print(finished)
52         if msg and started:
53             if msg & 1:
54                 delay = 0.1 * random() + 0.2 + M
55             else:
56                 delay = random() * 0.9 * M
57             msg >>= 1
58
59             if q.empty():
60                 data = urandom(randint(1, 100))
61                 data = len(data).to_bytes(4, "big") + data
62             else:
63                 data = q.get()
64
65             error(colored(f"PROXY::sending data {data}", "blue"))
66             sleep(delay)
67             server.send(data)
68         elif not q.empty():
69             if started:
70                 data = q.get()
71                 server.send(data)
72             else:
73                 server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
74                 error(colored(f"PROXY::trying {host}:{port}", "blue"))
75                 server.connect((host, port))
76
77                 started = True
78
79     server.close()
81
82 finished = False
83 receive = Thread(target=listen_and_buffer, args=(port,))
84 receive.start()
85 send = Thread(
86     target=be_proxy_and_send_message, args=(server_host, server_port, msg)
87 )
88 send.start()

```

```

90     if __name__ == "__main__":
91         parser = argparse.ArgumentParser()
92         parser.add_argument(
93             "-p", "--port", help="proxy running port", type=int, required=True
94         )
95         parser.add_argument(
96             "--server-port", help="server running port", type=int, required=True
97         )
98         parser.add_argument(
99             "--server-host", help="server running host", type=str, required=True
100        )
101        parser.add_argument(
102            "-m", "--message", help="secret message", type=str, required=True
103        )
104        args = parser.parse_args()
105
106        start_proxy(args.port, args.server_host, args.server_port, args.message)

```

proxy2.py

```

17     M = 1.0
18
19
20     def start_proxy(port: int, server_host: str, server_port: int, mode: bool):
21         q = Queue()
22
23     def listen_and_buffer(port):
24         error(colored(f"PROXY::binding to localhost:{port}", "red"))
25         sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
26         sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
27         sock.bind(("localhost", port))
28         sock.listen(1)
29
30         client, addr = sock.accept()
31         error(colored(f"PROXY::Connection from {addr}", "red"))
32
33         start = None
34         while True:
35             data = client.recv(1024)
36             end = time()
37             if start is None:
38                 data = (data, 0.0)
39             else:
40                 data = (data, end - start)
41                 end = start
42
43             error(colored(f"PROXY::Received data {data}", "red"))
44             q.put(data)
45             if not data[0]:
46                 break
47             client.close()
48
49         receive = Thread(target=listen_and_buffer, args=(port,))
50         receive.start()
51

```

```

52     if mode:
53         while True:
54             if not q.empty():
55                 server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
56                 error(colored(f"PROXY::trying {server_host}:{server_port}", "blue"))
57                 server.connect((server_host, server_port))
58
59                 while True:
60                     if q.empty():
61                         continue
62
63                     data, delay = q.get()
64                     if data == b"":
65                         break
66
67                     sleep(M - delay)
68                     error(colored(f"PROXY::sending data {data}", "green"))
69                     server.send(data)
70
71                 server.close()
72             server.close()
73     else:
74         while True:
75             if not q.empty():
76                 server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
77                 error(colored(f"PROXY::trying {server_host}:{port}", "blue"))
78                 server.connect((server_host, server_port))
79
80                 while True:
81                     if q.empty():
82                         continue
83
84                     data, _ = q.get()
85
86                     if data == b"":
87                         break
88
89                     error(colored(f"PROXY::sending data {data}", "green"))
90                     server.send(data)
91                     if random() < 0.2:
92                         data = urandom(randint(1, 10))
93                         data = len(data).to_bytes(4, "big") + data
94
95                     sleep(random())
96                     error(colored(f"PROXY::sending data {data}", "white"))
97                     server.send(data)
98
99                 server.close()

```

Демонстрация работы закладки и защиты

<pre>sarkoxedaf@herogasmotron2 ~/Working/University/covert_channels/lab3 [master] \$ % python user1/client.py -p 7777 --host localhost !!1118</pre>	<pre>sarkoxedaf@herogasmotron2 ~/Working/University/covert_channels/lab3 [master] \$ % python user2/server.py --port 9999 !!1184 ERROR:root:SERVER::binding to localhost:9999 ERROR:root:SERVER::Connection from ('127.0.0.1', 42046) ERROR:root:SERVER::recieved data b'\x00\x00\x00\x01\x89'</pre>
<pre>sarkoxedaf@herogasmotron2 ~/Working/University/covert_channels/lab3 [master] \$ % python user2/proxy.py --port 7777 --server-host localhost --server-port 8887 --message "Hid this, check out!!" ERROR:root:PROXY::binding to localhost:7777 ERROR:root:PROXY::Connection from ('127.0.0.1', 52206) ERROR:root:PROXY::Received data b'\x00\x00\x00\x01\x89' ERROR:root:PROXY::trying localhost:8887 ERROR:root:PROXY::sending data b'\x00\x00\x00\x01\x89' ERROR:root:PROXY::Received data b'\x00\x00\x00\n\xda:\xc6\xe6' s(z' ERROR:root:PROXY::sending data b'\x00\x00\x00\n\xda:\xc6\xe6' s(z' ERROR:root:PROXY::Received data b'\x00\x00\x00\x05\x0tQm\xf4' ERROR:root:PROXY::sending data b'\x00\x00\x00\x05\x0tQm\xf4' ERROR:root:PROXY::Received data b'\x00\x00\x00\x03a\xe7\xfe' ERROR:root:PROXY::sending data b'\x00\x00\x00\x03a\xe7\xfe' ERROR:root:PROXY::sending data b'\x00\x00\x00\x03\x8c\xdfV' ERROR:root:PROXY::sending data b'\x00\x00\x00\x02\xe5xc5'</pre>	<pre>sarkoxedaf@herogasmotron2 ~/Working/University/covert_channels/lab3 [master] \$ % python user2/proxy.py --port 8887 --server-host localhost --server-port 9999 --mode 1 ERROR:root:PROXY::binding to localhost:8887 ERROR:root:PROXY::Connection from ('127.0.0.1', 47060) ERROR:root:PROXY::Received data (b'\x00\x00\x00\x01\x89', 0.0) ERROR:root:PROXY::trying localhost:9999 ERROR:root:PROXY::Received data (b'\x00\x00\x00\n\xda:\xc6\xe6' s(z', 0.0) ERROR:root:PROXY::Received data (b'\x00\x00\x00\x05\x0tQm\xf4', 0.0) ERROR:root:PROXY::Received data (b'\x00\x00\x00\x03a\xe7\xfe', 0.0) ERROR:root:PROXY::Received data (b'\x00\x00\x00\x03\x8c\xdfV', 0.0) ERROR:root:PROXY::sending data b'\x00\x00\x00\x01\x89'</pre>
<pre>sarkoxedaf@herogasmotron2 ~/Working/University/covert_channels/lab3 [master] \$ % python user1/client.py -p 7777 --host localhost !!11212</pre>	<pre>sarkoxedaf@herogasmotron2 ~/Working/University/covert_channels/lab3 [master] \$ % python user2/server.py --port 9999 !!11205 ERROR:root:SERVER::binding to localhost:9999 ERROR:root:SERVER::Connection from ('127.0.0.1', 59728) ERROR:root:SERVER::recieved data b'\x00\x00\x00\x06\xb2\xd8Z!D\x04' ERROR:root:SERVER::recieved data b'\x00\x00\x00\x02\x10\xf3'</pre>
<pre>sarkoxedaf@herogasmotron2 ~/Working/University/covert_channels/lab3 [master] \$ % python user2/proxy.py --port 7777 --server-host localhost --server-port 8887 --message "Hid this, check out!!" ERROR:root:PROXY::binding to localhost:7777 ERROR:root:PROXY::Connection from ('127.0.0.1', 35124) ERROR:root:PROXY::Received data b'\x00\x00\x00\x06\xb2\xd8Z!D\x04' ERROR:root:PROXY::trying localhost:8887 ERROR:root:PROXY::sending data b'\x00\x00\x00\x06\xb2\xd8Z!D\x04' ERROR:root:PROXY::Received data b'\x00\x00\x00\x02\x10\xf3' ERROR:root:PROXY::sending data b'\x00\x00\x00\x02\x10\xf3' ERROR:root:PROXY::sending data b'\x00\x00\x00\x05\x0f\x1c\xel9\xdc' ERROR:root:PROXY::Received data b'\x00\x00\x00\nm\x88\xf6\x15\x00\xc7\x84' ERROR:root:PROXY::sending data b'\x00\x00\x00\x02\xe5xc5'</pre>	<pre>sarkoxedaf@herogasmotron2 ~/Working/University/covert_channels/lab3 [master] \$ % python user2/proxy.py --port 8887 --server-host localhost --server-port 9999 --mode 0 ERROR:root:PROXY::binding to localhost:8887 ERROR:root:PROXY::Connection from ('127.0.0.1', 57522) ERROR:root:PROXY::Received data (b'\x00\x00\x00\x06\xb2\xd8Z!D\x04', 0.0) ERROR:root:PROXY::trying localhost:8887 ERROR:root:PROXY::sending data b'\x00\x00\x00\x06\xb2\xd8Z!D\x04' ERROR:root:PROXY::Received data (b'\x00\x00\x00\x02\x10\xf3', 0.0) ERROR:root:PROXY::sending data b'\x00\x00\x00\x02\x10\xf3'</pre>
<pre>[0] 0:python*</pre>	<pre>"herogasmotron2" 19:35 21-Mar-2024</pre>

Демонстрация работы злоумышленника

Нормализация

```
, 1.0002222061157227, 1.00020432472229, 1.000230073928833, 1.000326395034  
79, 1.001526117324829, 1.002342700958252, 1.0001840591430664, 1.000221729  
2785645, 1.0002331733703613, 1.0002286434173584, 1.0002477169036865, 1.00  
06024837493896, 1.0001778602600098, 1.0002410411834717, 1.000221967697143  
6, 1.0002264976501465, 1.000230312347412, 1.00193190574646, 1.00085997581  
48193, 1.000464677810669, 1.0000851154327393, 1.000291109085083, 1.000201  
4636993408, 1.0002596378326416, 1.0002098083496094, 2.6226043701171875e-0  
5]
```

```
ERROR:root:Recovered message: b'?\xff\xff\xff\xff\xff\xff\xff\xff\xff  
\xff\xff\xff\xff\xff\xff\xff\xff\xff'
```

□

Фиктивные пакеты:

```
43800354004, 0.762732744216919, 0.36688804626464844, 0.7343463897705078,  
0.555079460144043, 0.1961348056793213, 0.7109658718109131, 0.215986013412  
47559, 0.029045820236206055, 0.10543298721313477, 0.37146997451782227, 0.  
38469600677490234, 0.2131969928741455, 0.3140983581542969, 0.786209583282  
4707, 0.25713038444519043, 0.11668133735656738, 0.06682085990905762, 0.16  
183876991271973, 0.3476269245147705, 0.34636926651000977, 0.7791616916656  
494, 0.009315013885498047, 0.03346824645996094, 0.7423806190490723, 0.761  
2903118133545, 0.38947391510009766, 0.7275714874267578, 0.944987773895263  
7, 2.47955322265625e-05, 1.7642974853515625e-05, 0.24085474014282227, 2.5  
033950805664062e-05, 0.03184056282043457, 0.759533166885376, 0.7705786228  
179932, 0.25400829315185547, 0.0447080135345459, 0.06457138061523438, 0.0  
2575516700744629, 0.7168200016021729, 0.26305413246154785, 0.617027044296  
2646, 2.47955322265625e-05, 0.47793102264404297, 0.001234292984008789]
```

```
ERROR:root:Recovered message: b'\x01C\x06\xc8\x10\x16\x8cF\x8b\x99P\x80\x  
9aXFi3D\x81\xbd\xc6\xe8\x10'
```

Выводы

Нормализация

Пропускная способность скрытого канала падает, так как от среднего в пол секунды промежутка мы переходим к одной секунде всегда.

Аналогично с основным каналом, так как там накладывается еще и задержка, создаваемая закладкой и фиктивный трафик.

Однако схема полностью справилась с противодействием скрытому каналу, так как теперь медианные значения всегда будут давать 1

Фиктивный трафик

Пропускная способность скрытого канала изменяется, так как мы добавляем случайные задержки и дополнительные пакеты, которые сильно влияют на возможность декодирования.

Пропускная способность основного канала изменяется значительно, так как у нас появляется уже удвоенный фиктивный трафик и задержки.

Схема полностью справилась с противодействием скрытому каналу, так как теперь у нас не сохраняется как последовательность бит, так и временные отрезки, необходимые для декодирования.