

Cache Template Attacks:

Automating Attacks on Inclusive Last-Level Caches

Daniel Gruss, Raphael Spreitzer, and Stefan Mangard,
IAIK, Graz University of Technology

August 14, 2015

Cache Template Attacks

Cache Template Attacks

- State of the art: cache attacks are powerful
- Problem: manual identification of attack targets

Cache Template Attacks

- State of the art: cache attacks are powerful
- Problem: manual identification of attack targets
- **Solution: Cache Template Attacks**
- Automatically find any secret-dependent cache access
- Can be used for attacks and to improve software

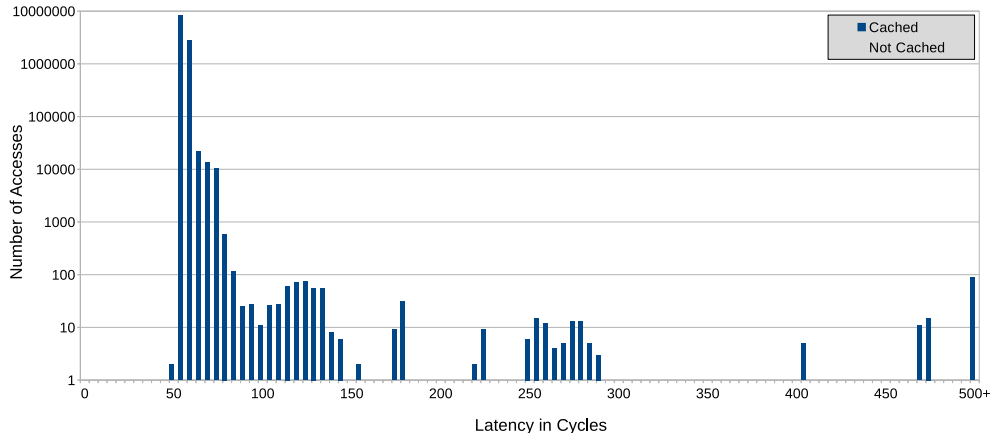
Cache Template Attacks

- State of the art: cache attacks are powerful
- Problem: manual identification of attack targets
- **Solution: Cache Template Attacks**
- Automatically find any secret-dependent cache access
- Can be used for attacks and to improve software
- Examples:
 - Cache-based keylogger
 - Automatic attacks on crypto algorithms

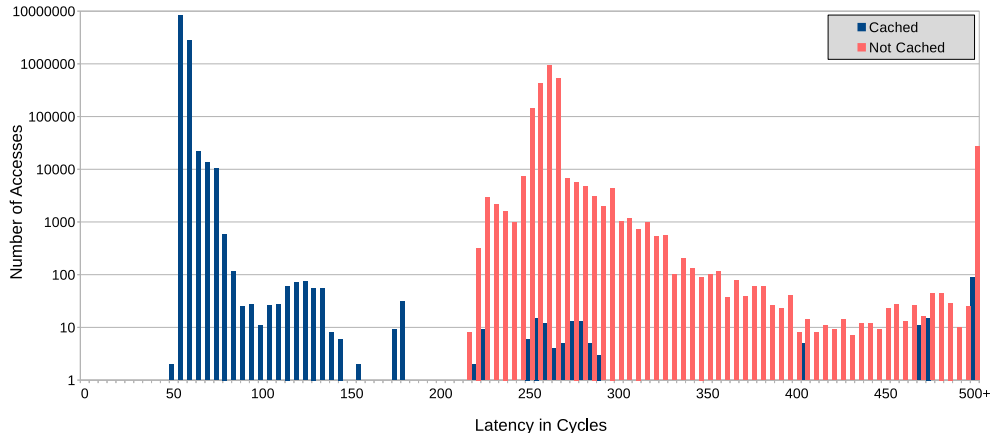
CPU Caches

- Reduce memory access latency

Memory Access Latency



Memory Access Latency



CPU Caches

- Reduce memory access latency
- Last-level cache on modern Intel CPUs:
 - Inclusive (to lower levels)
 - Not in Last-level cache $\Leftrightarrow$ not cached
 - Shared (between cores)

Flush+Reload

- Powerful cache attack
- Works on shared binaries/libraries
- Application on crypto algorithms

Flush+Reload

Attacker address space



Cache



Victim address space



Cache is empty

Flush+Reload

Attacker address space



Cache



Victim address space



Victim maps shared library

Flush+Reload

Attacker address space



Cache



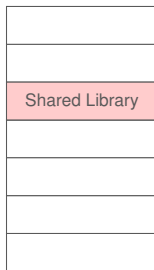
Victim address space



Attacker maps shared library

Flush+Reload

Attacker address space



Cache



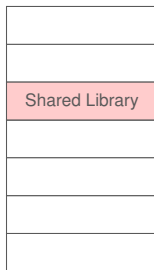
Victim address space



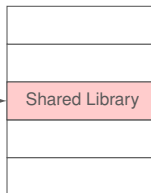
Attacker accesses shared library

Flush+Reload

Attacker address space



Cache



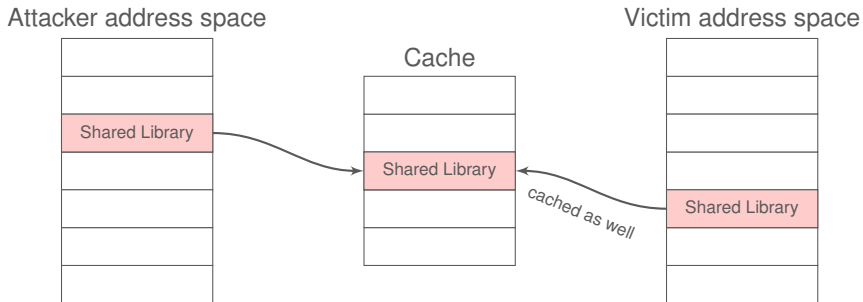
load

Victim address space



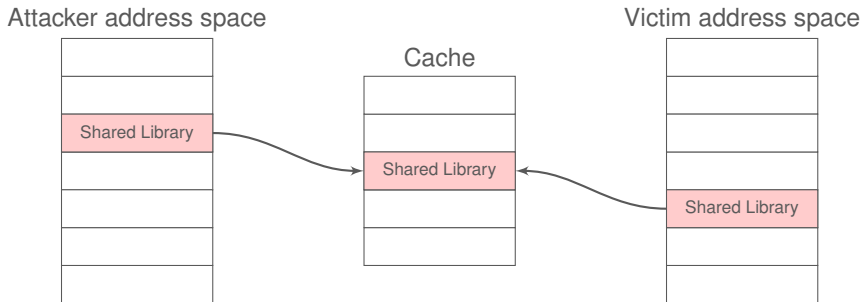
Loading into cache...

Flush+Reload



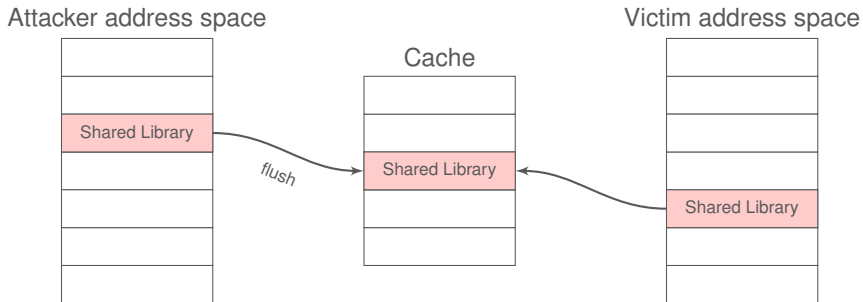
Loading into cache...

Flush+Reload



Attacker measures high latency

Flush+Reload



Attacker flushes shared library (“flush”)

Flush+Reload

Attacker address space



Cache

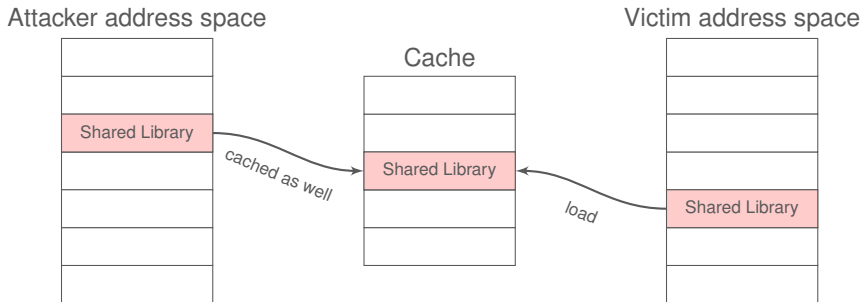


Victim address space



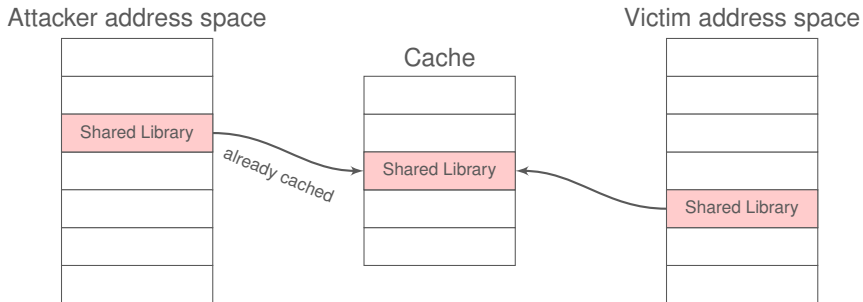
Cache is empty again

Flush+Reload



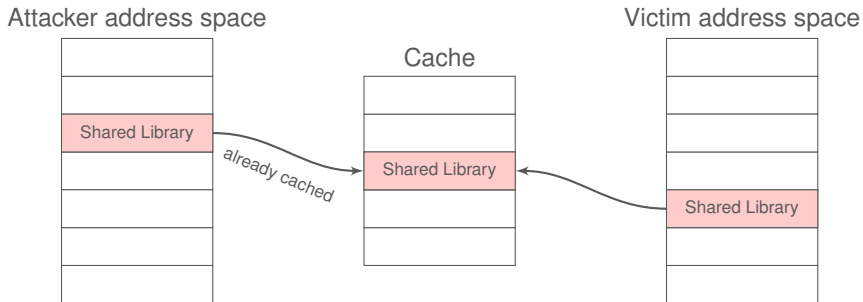
Victim accesses shared library

Flush+Reload



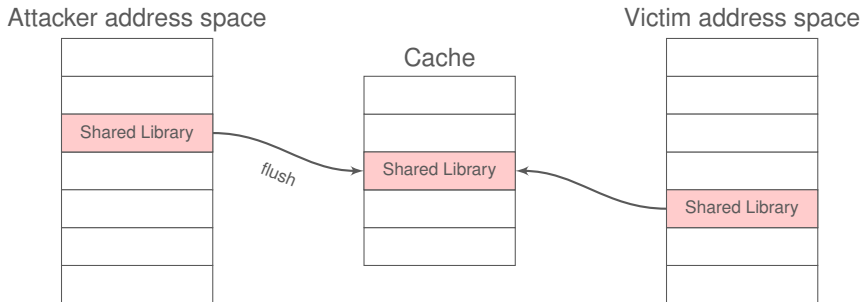
Attacker accesses shared library ("reload")

Flush+Reload



Attacker measures low latency

Flush+Reload



Attacker flushes shared library (“flush”)

Flush+Reload

Attacker address space



Cache

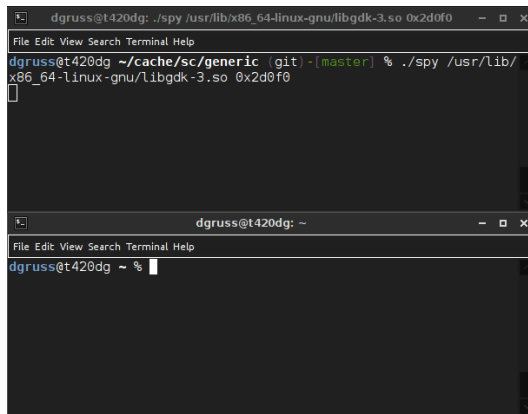


Victim address space



Cache is empty again

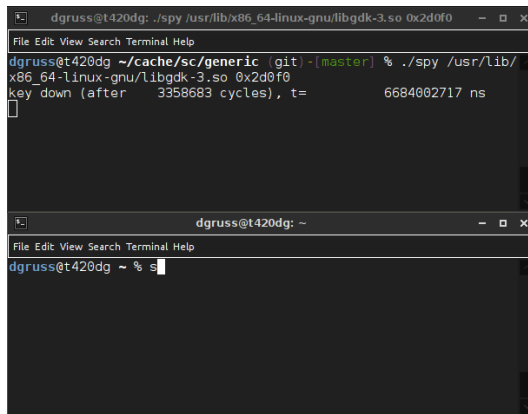
Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window has a title bar 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and a menu bar 'File Edit View Search Terminal Help'. The prompt is 'dgruss@t420dg ~/cache/sc/generic (git) -[master] %' and the command entered is './spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0'. The bottom window has a title bar 'dgruss@t420dg: ~' and a menu bar 'File Edit View Search Terminal Help'. The prompt is 'dgruss@t420dg ~ %' and the command entered is '%'. Both windows have a dark background and a light cursor.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
dgruss@t420dg ~/cache/sc/generic (git) -[master] % ./spy /usr/lib/
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ %
```

Can We Build a Cache-Based Keylogger?

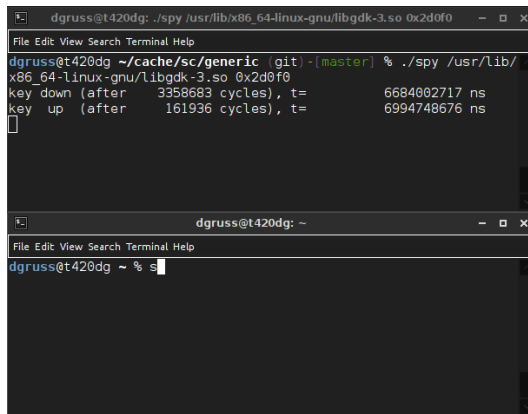


The image shows two terminal windows. The top window has a title bar 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and a menu bar 'File Edit View Search Terminal Help'. The prompt is 'dgruss@t420dg ~/cache/sc/generic (git) -[master] %'. The command './spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' has been executed, resulting in the output 'key_down (after 3358683 cycles), t= 6684002717 ns'. The bottom window has a title bar 'dgruss@t420dg: ~' and a menu bar 'File Edit View Search Terminal Help'. The prompt is 'dgruss@t420dg ~ %' and the command 's' has been entered.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
dgruss@t420dg ~/cache/sc/generic (git) -[master] % ./spy /usr/lib/
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
key_down (after 3358683 cycles), t= 6684002717 ns
█

dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % s█
```

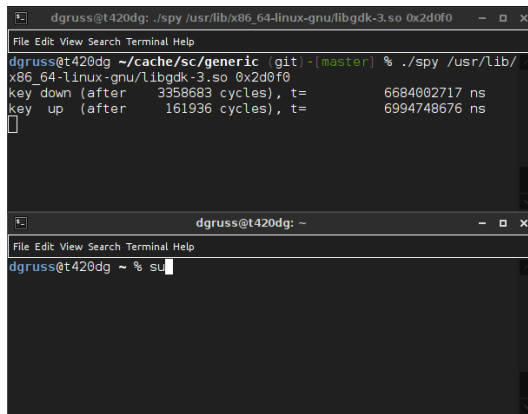
Can We Build a Cache-Based Keylogger?



```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
dgruss@t420dg ~/cache/sc/generic (git) -[master] % ./spy /usr/lib/
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
key down (after 3358683 cycles), t= 6684002717 ns
key up (after 161936 cycles), t= 6994748676 ns
█

dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % s█
```

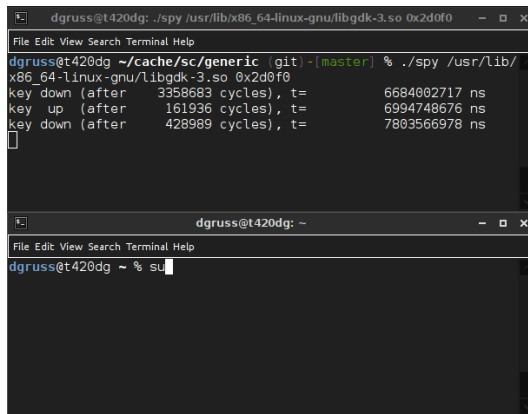
Can We Build a Cache-Based Keylogger?



```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
dgruss@t420dg ~/cache/sc/generic (git) -[master] % ./spy /usr/lib/
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
key down (after 3358683 cycles), t= 6684002717 ns
key up (after 161936 cycles), t= 6994748676 ns
█

dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % su█
```

Can We Build a Cache-Based Keylogger?



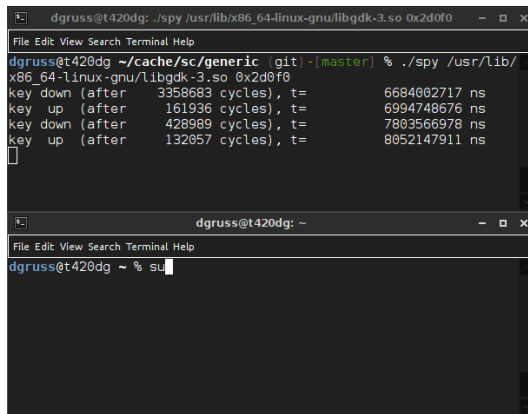
The image shows two terminal windows. The top window has a title bar 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and a menu bar 'File Edit View Search Terminal Help'. The prompt is 'dgruss@t420dg ~/cache/sc/generic (git) -[master] %'. The command executed is './spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0'. The output shows three key events with their cycle counts and timestamps in nanoseconds.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
dgruss@t420dg ~/cache/sc/generic (git) -[master] % ./spy /usr/lib/
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
key_down (after 3358683 cycles), t= 6684002717 ns
key_up (after 161936 cycles), t= 6994748676 ns
key_down (after 428989 cycles), t= 7803566978 ns
█
```

The bottom window has a title bar 'dgruss@t420dg: ~' and a menu bar 'File Edit View Search Terminal Help'. The prompt is 'dgruss@t420dg ~ %'. The command executed is 'su'.

```
dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % su█
```

Can We Build a Cache-Based Keylogger?

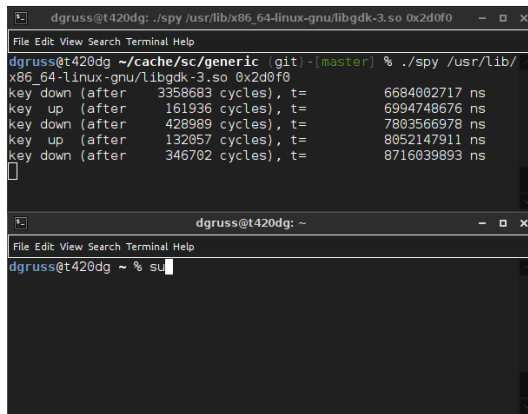


The image shows two terminal windows. The top window is titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and displays the output of a program that logs key events. The bottom window is titled 'dgruss@t420dg: ~' and shows a shell prompt with 'su' entered.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
dgruss@t420dg ~/cache/sc/generic (git) -[master] % ./spy /usr/lib/
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
key_down (after 3358683 cycles), t= 6684002717 ns
key_up (after 161936 cycles), t= 6994748676 ns
key_down (after 428989 cycles), t= 7803566978 ns
key_up (after 132057 cycles), t= 8052147911 ns
█

dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % su█
```

Can We Build a Cache-Based Keylogger?

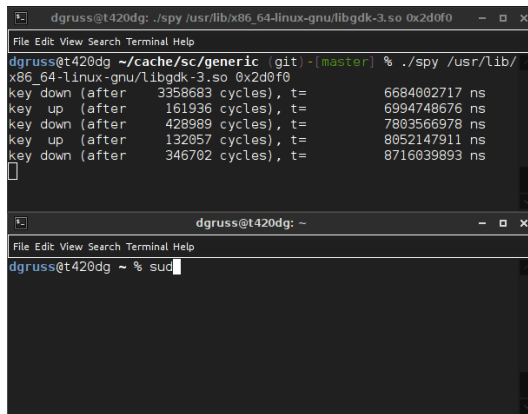


The image shows two terminal windows. The top window is titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and displays the output of a program that monitors key presses. The output shows five key events with their respective cycle counts and timestamps. The bottom window is titled 'dgruss@t420dg: ~' and shows the command 'su' being entered, indicating a switch to root privileges.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
dgruss@t420dg ~/cache/sc/generic (git) -[master] % ./spy /usr/lib/
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
key_down (after 3358683 cycles), t= 6684002717 ns
key_up (after 161936 cycles), t= 6994748676 ns
key_down (after 428989 cycles), t= 7803566978 ns
key_up (after 132057 cycles), t= 8052147911 ns
key_down (after 346702 cycles), t= 8716039893 ns
█

dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % su█
```

Can We Build a Cache-Based Keylogger?



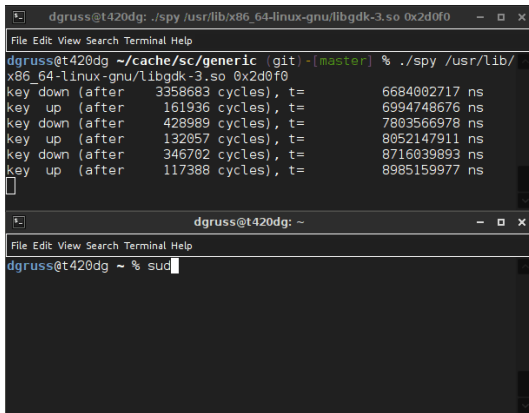
The image shows two terminal windows. The top window is titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and displays the output of a program that monitors key presses. The output shows five key events with their respective cycle counts and timestamps. The bottom window is titled 'dgruss@t420dg: ~' and shows the command 'sudo' being entered.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
dgruss@t420dg ~/cache/sc/generic (git) -[master] % ./spy /usr/lib/
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
key_down (after 3358683 cycles), t= 6684002717 ns
key_up (after 161936 cycles), t= 6994748676 ns
key_down (after 428989 cycles), t= 7803566978 ns
key_up (after 132057 cycles), t= 8052147911 ns
key_down (after 346702 cycles), t= 8716039893 ns

```

```
dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo
```

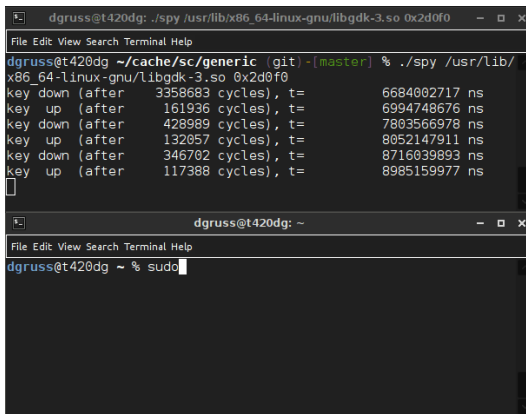
Can We Build a Cache-Based Keylogger?



```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
dgruss@t420dg ~/cache/sc/generic (git)-[master] % ./spy /usr/lib/
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
key_down (after 3358683 cycles), t= 6684002717 ns
key_up (after 161936 cycles), t= 6994748676 ns
key_down (after 428989 cycles), t= 7803566978 ns
key_up (after 132057 cycles), t= 8052147911 ns
key_down (after 346702 cycles), t= 8716039893 ns
key_up (after 117388 cycles), t= 8985159977 ns
█

dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo █
```

Can We Build a Cache-Based Keylogger?



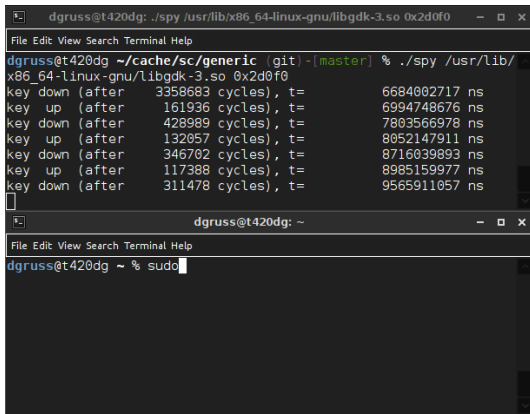
The image shows two terminal windows. The top window is titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0'. It shows the execution of a program that logs key events. The output is as follows:

```
dgruss@t420dg ~/cache/sc/generic (git)-[master] % ./spy /usr/lib/
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
key_down (after 3358683 cycles), t= 6684002717 ns
key_up (after 161936 cycles), t= 6994748676 ns
key_down (after 428989 cycles), t= 7803566978 ns
key_up (after 132057 cycles), t= 8052147911 ns
key_down (after 346702 cycles), t= 8716039893 ns
key_up (after 117388 cycles), t= 8985159977 ns
```

The bottom window is titled 'dgruss@t420dg: ~'. It shows the user typing the command 'sudo' followed by a cursor.

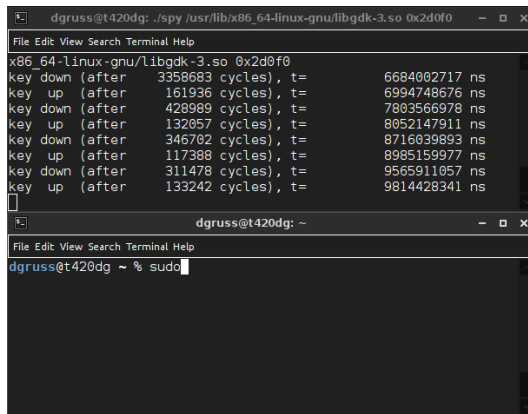
```
dgruss@t420dg ~ % sudo
```

Can We Build a Cache-Based Keylogger?



```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
dgruss@t420dg ~/cache/sc/generic (git) -[master] % ./spy /usr/lib/
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
key_down (after 3358683 cycles), t= 6684002717 ns
key_up (after 161936 cycles), t= 6994748676 ns
key_down (after 428989 cycles), t= 7803566978 ns
key_up (after 132057 cycles), t= 8052147911 ns
key_down (after 346702 cycles), t= 8716039893 ns
key_up (after 117388 cycles), t= 8985159977 ns
key_down (after 311478 cycles), t= 9565911057 ns
dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo
```

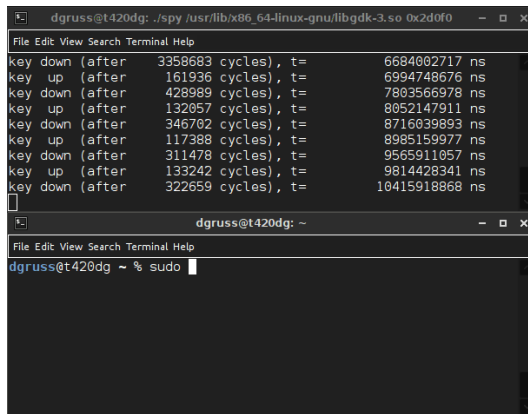
Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window has a title bar 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0'. It displays the output of a program that logs key events and their timing. The output shows a sequence of key presses and releases with their respective cycle counts and timestamps in nanoseconds. The bottom window has a title bar 'dgruss@t420dg: ~' and shows the command 'sudo' being entered at the prompt.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
x86_64-linux-gnu/libgdk-3.so 0x2d0f0
key down (after 3358683 cycles), t= 6684002717 ns
key up (after 161936 cycles), t= 6994748676 ns
key down (after 428989 cycles), t= 7803566978 ns
key up (after 132057 cycles), t= 8052147911 ns
key down (after 346702 cycles), t= 8716039893 ns
key up (after 117388 cycles), t= 8985159977 ns
key down (after 311478 cycles), t= 9565911057 ns
key up (after 133242 cycles), t= 9814428341 ns
dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo
```

Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window displays the output of a program that logs keypresses and the time taken for each. The bottom window shows a prompt for a sudo command.

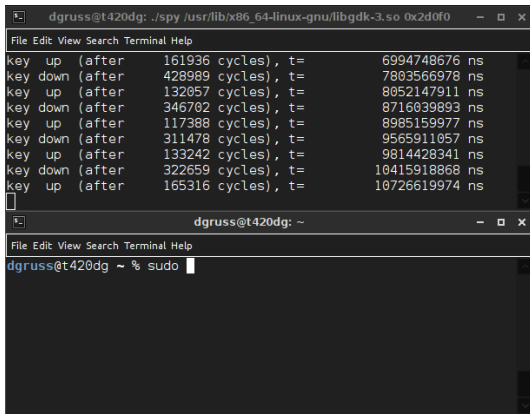
```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
```

Event	Cycles	Time (ns)
key down (after 3358683 cycles), t=	3358683	6684002717
key up (after 161936 cycles), t=	161936	6994748676
key down (after 428989 cycles), t=	428989	7803566978
key up (after 132057 cycles), t=	132057	8052147911
key down (after 346702 cycles), t=	346702	8716039893
key up (after 117388 cycles), t=	117388	8985159977
key down (after 311478 cycles), t=	311478	9565911057
key up (after 133242 cycles), t=	133242	9814428341
key down (after 322659 cycles), t=	322659	10415918868

```
dgruss@t420dg: ~
```

```
dgruss@t420dg ~ % sudo
```

Can We Build a Cache-Based Keylogger?

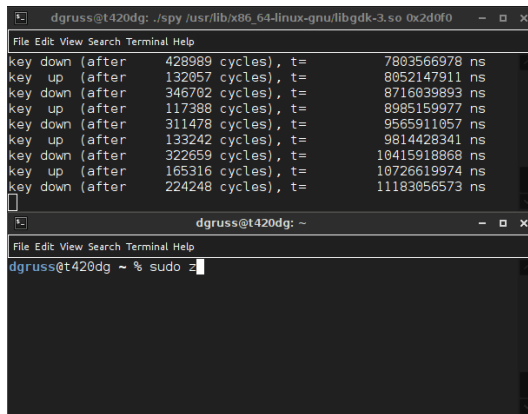


The image shows two terminal windows. The top window displays the output of a program that logs keypresses and the time taken for each. The output is as follows:

Event	Cycles	Time (ns)
key up	161936	6994748676
key down	428989	7803566978
key up	132057	8052147911
key down	346702	8716039893
key up	117388	8985159977
key down	311478	9565911057
key up	133242	9814428341
key down	322659	10415918868
key up	165316	10726619974

The bottom window shows the user prompt `dgruss@t420dg: ~` and the command `sudo` being entered.

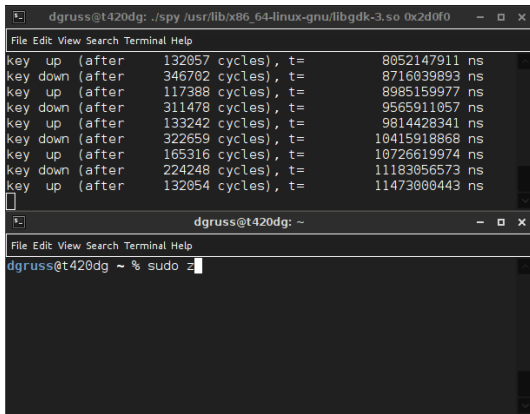
Can We Build a Cache-Based Keylogger?



```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key down (after 428989 cycles), t= 7803566978 ns
key up (after 132057 cycles), t= 8052147911 ns
key down (after 346702 cycles), t= 8716039893 ns
key up (after 117388 cycles), t= 8985159977 ns
key down (after 311478 cycles), t= 9565911057 ns
key up (after 133242 cycles), t= 9814428341 ns
key down (after 322659 cycles), t= 10415918868 ns
key up (after 165316 cycles), t= 10726619974 ns
key down (after 224248 cycles), t= 11183056573 ns
█

dgruss@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo z█
```

Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window displays the output of a program that logs keypresses and the time taken for each. The bottom window shows a prompt for a command.

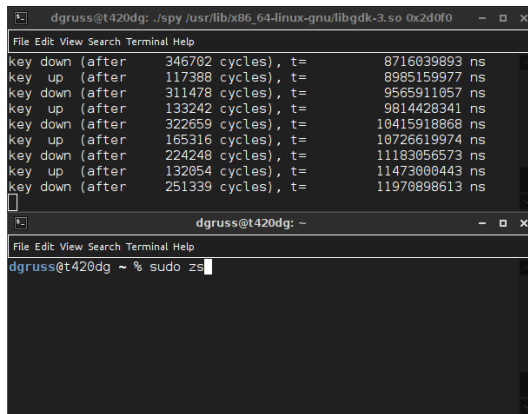
```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
```

Key	Time (ns)
key up (after 132057 cycles), t=	8052147911 ns
key down (after 346702 cycles), t=	8716039893 ns
key up (after 117388 cycles), t=	8985159977 ns
key down (after 311478 cycles), t=	9565911057 ns
key up (after 133242 cycles), t=	9814428341 ns
key down (after 322659 cycles), t=	10415918868 ns
key up (after 165316 cycles), t=	10726619974 ns
key down (after 224248 cycles), t=	11183056573 ns
key up (after 132054 cycles), t=	11473000443 ns

```
dgruss@t420dg: ~
```

```
dgruss@t420dg ~ % sudo z
```

Can We Build a Cache-Based Keylogger?

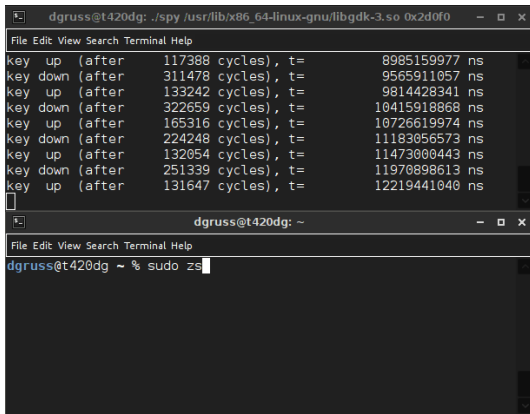


The screenshot shows a terminal window with a title bar indicating the command `./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0`. The terminal output displays a series of key events with timestamps in nanoseconds (ns). The events are as follows:

Event	Cycles	Time (ns)
key down	346702	8716039893
key up	117388	8985159977
key down	311478	9565911057
key up	133242	9814428341
key down	322659	10415918868
key up	165316	10726619974
key down	224248	11183056573
key up	132054	11473000443
key down	251339	11970898613

Below the output, the terminal shows a shell prompt `dgruss@t420dg: ~` and the command `sudo zs` being entered.

Can We Build a Cache-Based Keylogger?

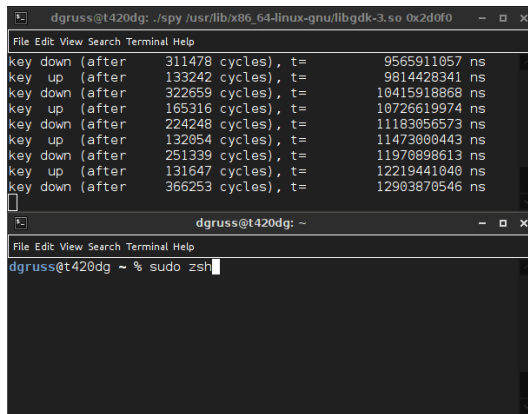


The screenshot shows a terminal window with a title bar indicating the command `./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0`. The terminal output displays a series of key events with their corresponding cycle counts and timestamps in nanoseconds (ns). The events are as follows:

Event	Cycles	Timestamp (ns)
key up	117388	8985159977
key down	311478	9565911057
key up	133242	9814428341
key down	322659	10415918868
key up	165316	10726619974
key down	224248	11183056573
key up	132054	11473000443
key down	251339	11970898613
key up	131647	12219441040

Below the first window, another terminal window is visible with the prompt `dgruss@t420dg: ~` and the command `sudo zs` being entered.

Can We Build a Cache-Based Keylogger?

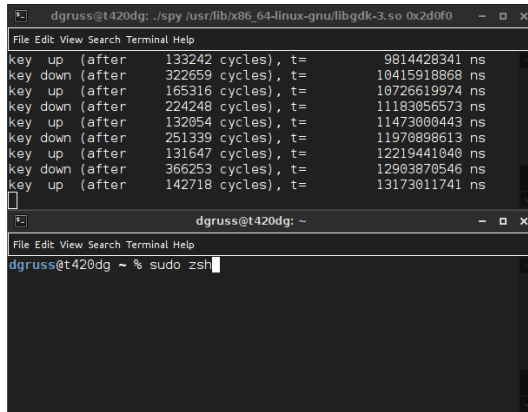


The image shows a terminal window with a dark background. The title bar reads 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0'. The terminal content displays a series of key events with timestamps in nanoseconds (ns). The events are as follows:

Event	Cycles	Time (ns)
key down	311478	9565911057
key up	133242	9814428341
key down	322659	10415918868
key up	165316	10726619974
key down	224248	11183056573
key up	132054	11473000443
key down	251339	11970898613
key up	131647	12219441040
key down	366253	12903870546

Below the table, the terminal shows a shell prompt 'dgruss@t420dg: ~' and the command 'dgruss@t420dg ~ % sudo zsh' being entered.

Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window displays the output of a program that logs keypresses, showing the time taken for each key to be pressed and released. The bottom window shows a shell prompt where the user has entered the command 'sudo zsh'.

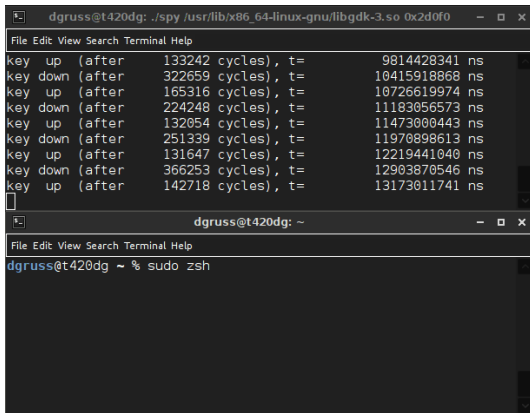
```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
```

Key	Time (ns)
key up (after 133242 cycles), t=	9814428341 ns
key down (after 322659 cycles), t=	10415918868 ns
key up (after 165316 cycles), t=	10726619974 ns
key down (after 224248 cycles), t=	11183056573 ns
key up (after 132054 cycles), t=	11473000443 ns
key down (after 251339 cycles), t=	11970898613 ns
key up (after 131647 cycles), t=	12219441040 ns
key down (after 366253 cycles), t=	12903870546 ns
key up (after 142718 cycles), t=	13173011741 ns

```
dgruss@t420dg: ~
```

```
dgruss@t420dg ~ % sudo zsh
```

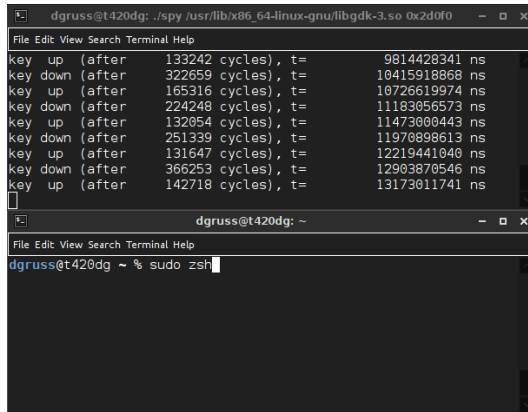
Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window has a title bar 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and a menu bar 'File Edit View Search Terminal Help'. It displays a list of key events with their cycle counts and timestamps in nanoseconds. The bottom window has a title bar 'dgruss@t420dg: ~' and a menu bar 'File Edit View Search Terminal Help'. It shows the command 'dgruss@t420dg ~ % sudo zsh'.

Event	Cycles	Timestamp (ns)
key up	133242	9814428341
key down	322659	10415918868
key up	165316	10726619974
key down	224248	11183056573
key up	132054	11473000443
key down	251339	11970898613
key up	131647	12219441040
key down	366253	12903870546
key up	142718	13173011741

Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window displays the output of a program that logs keypresses and the time taken for the CPU cache to refill. The data shows that keypresses occur at regular intervals, with the time between key events being approximately 130,000 to 140,000 cycles (or 100 to 110 ns). The bottom window shows a prompt for a shell command.

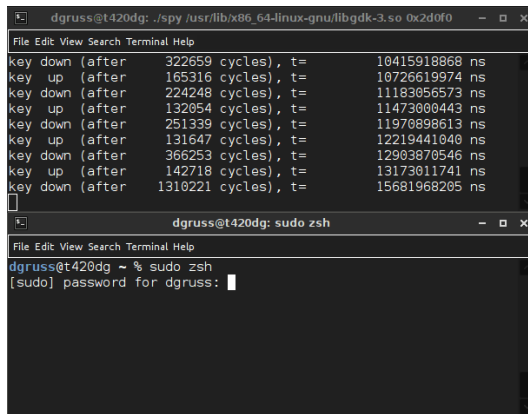
```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
```

Event	Cycles	Time (ns)
key up (after 133242 cycles)	133242	9814428341 ns
key down (after 322659 cycles)	322659	10415918868 ns
key up (after 165316 cycles)	165316	10726619974 ns
key down (after 224248 cycles)	224248	11183056573 ns
key up (after 132054 cycles)	132054	11473000443 ns
key down (after 251339 cycles)	251339	11970898613 ns
key up (after 131647 cycles)	131647	12219441040 ns
key down (after 366253 cycles)	366253	12903870546 ns
key up (after 142718 cycles)	142718	13173011741 ns

```
dgruss@t420dg: ~
```

```
dgruss@t420dg ~ % sudo zsh
```

Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window displays a list of key events with their cycle counts and timestamps. The bottom window shows a prompt for the sudo password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
```

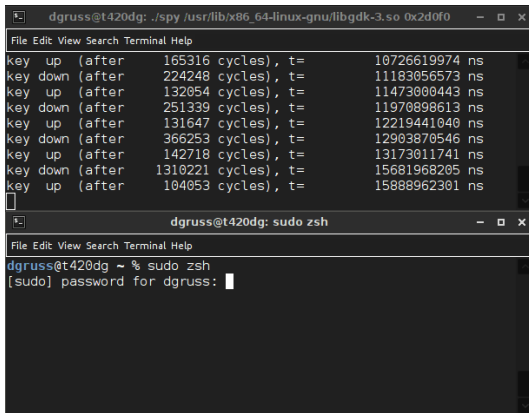
Event	Cycles	Timestamp (ns)
key down (after 322659 cycles), t=	322659	10415918868
key up (after 165316 cycles), t=	165316	10726619974
key down (after 224248 cycles), t=	224248	11183056573
key up (after 132054 cycles), t=	132054	11473000443
key down (after 251339 cycles), t=	251339	11970898613
key up (after 131647 cycles), t=	131647	12219441040
key down (after 366253 cycles), t=	366253	12903870546
key up (after 142718 cycles), t=	142718	13173011741
key down (after 1310221 cycles), t=	1310221	15681968205

```
dgruss@t420dg: sudo zsh
```

```
dgruss@t420dg ~ % sudo zsh
```

```
[sudo] password for dgruss:
```

Can We Build a Cache-Based Keylogger?

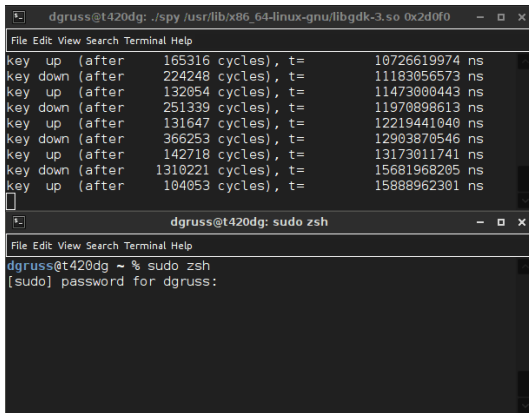


The image shows two terminal windows. The top window displays the output of a program running `./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0`. It shows a list of key events with timestamps in nanoseconds. The bottom window shows a `sudo zsh` prompt where the user's password is being requested.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 165316 cycles), t= 10726619974 ns
key down (after 224248 cycles), t= 11183056573 ns
key up (after 132054 cycles), t= 11473000443 ns
key down (after 251339 cycles), t= 11970898613 ns
key up (after 131647 cycles), t= 12219441040 ns
key down (after 366253 cycles), t= 12903870546 ns
key up (after 142718 cycles), t= 13173011741 ns
key down (after 1310221 cycles), t= 15681968205 ns
key up (after 104053 cycles), t= 15888962301 ns

dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
```

Can We Build a Cache-Based Keylogger?

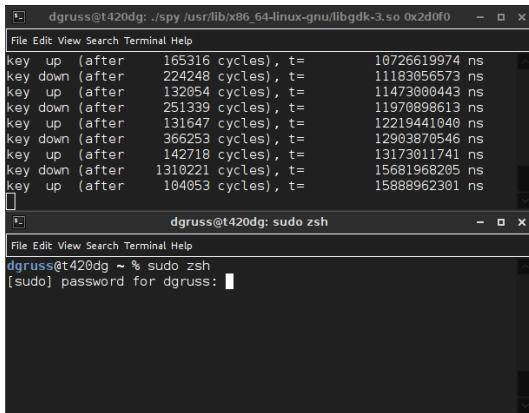


The image shows two terminal windows. The top window displays the output of a program running `./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0`. It shows a series of key events with timestamps in nanoseconds. The bottom window shows a `sudo zsh` prompt where the user's password is being requested.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 165316 cycles), t= 10726619974 ns
key down (after 224248 cycles), t= 11183056573 ns
key up (after 132054 cycles), t= 11473000443 ns
key down (after 251339 cycles), t= 11970898613 ns
key up (after 131647 cycles), t= 12219441040 ns
key down (after 366253 cycles), t= 12903870546 ns
key up (after 142718 cycles), t= 13173011741 ns
key down (after 1310221 cycles), t= 15681968205 ns
key up (after 104053 cycles), t= 15888962301 ns

dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
```

Can We Build a Cache-Based Keylogger?

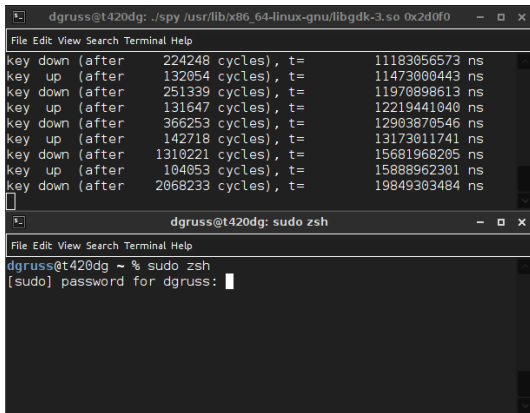


The image shows two terminal windows. The top window displays the output of a program running `./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0`. It shows a series of key events with timestamps in nanoseconds. The bottom window shows a `sudo zsh` prompt where the user's password is being requested.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 165316 cycles), t= 10726619974 ns
key down (after 224248 cycles), t= 11183056573 ns
key up (after 132054 cycles), t= 11473000443 ns
key down (after 251339 cycles), t= 11970898613 ns
key up (after 131647 cycles), t= 12219441040 ns
key down (after 366253 cycles), t= 12903870546 ns
key up (after 142718 cycles), t= 13173011741 ns
key down (after 1310221 cycles), t= 15681968205 ns
key up (after 104053 cycles), t= 15888962301 ns

dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
```

Can We Build a Cache-Based Keylogger?

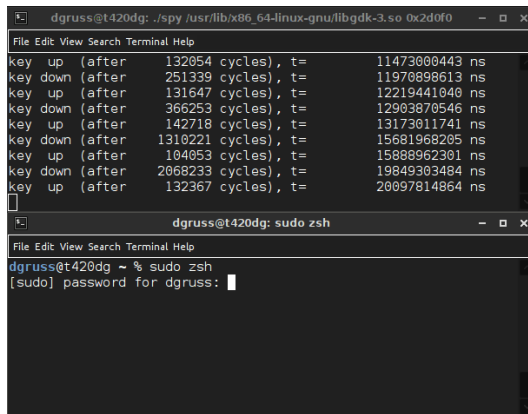


The image shows two terminal windows. The top window is titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and displays a list of key events with their cycle counts and timestamps. The bottom window is titled 'dgruss@t420dg: sudo zsh' and shows the prompt for the sudo password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key down (after 224248 cycles), t= 11183056573 ns
key up (after 132054 cycles), t= 11473000443 ns
key down (after 251339 cycles), t= 11970898613 ns
key up (after 131647 cycles), t= 12219441040 ns
key down (after 366253 cycles), t= 12903870546 ns
key up (after 142718 cycles), t= 13173011741 ns
key down (after 1310221 cycles), t= 15681968205 ns
key up (after 104053 cycles), t= 15888962301 ns
key down (after 2068233 cycles), t= 19849303484 ns
█

dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss: █
```

Can We Build a Cache-Based Keylogger?



The image displays two terminal windows. The top window shows the output of a program that logs keypresses, including the key state (up/down), the time in cycles, and the time in nanoseconds. The bottom window shows a prompt for the sudo password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
```

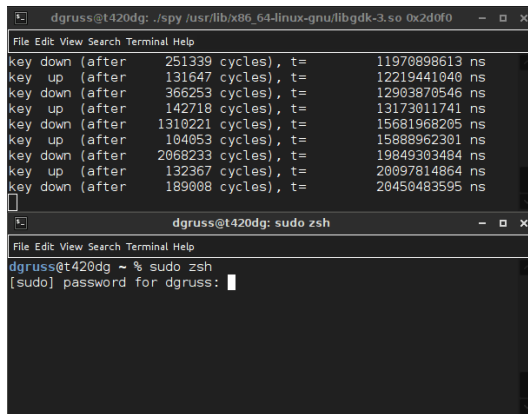
Key State	Cycles	Time (ns)
key up	132054	11473000443
key down	251339	11970898613
key up	131647	12219441040
key down	366253	12903870546
key up	142718	13173011741
key down	1310221	15681968205
key up	104053	15808962301
key down	2068233	19849303484
key up	132367	20097814864

```
dgruss@t420dg: sudo zsh
```

```
dgruss@t420dg ~ % sudo zsh
```

```
[sudo] password for dgruss:
```

Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window displays a list of key events with their cycle counts and timestamps. The bottom window shows a prompt for the sudo password.

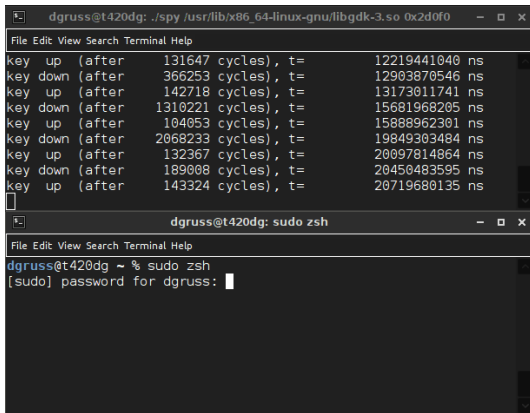
```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
```

Event	Cycles	Timestamp (ns)
key down	251339	11970898613
key up	131647	12219441040
key down	366253	12903870546
key up	142718	13173011741
key down	1310221	15681968205
key up	104053	15888962301
key down	2068233	19849303484
key up	132367	20097814864
key down	189008	20450483595

```
dgruss@t420dg: sudo zsh
```

```
[sudo] password for dgruss:
```

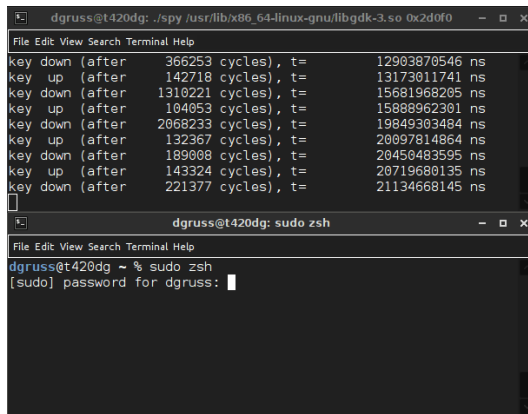
Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window, titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0', displays a list of key events with their cycle counts and timestamps. The bottom window, titled 'dgruss@t420dg: sudo zsh', shows the prompt '[sudo] password for dgruss:' with a cursor.

Event	Cycles	Timestamp (ns)
key up	131647	12219441040
key down	366253	12903870546
key up	142718	13173011741
key down	1310221	15681968205
key up	104053	15888962301
key down	2068233	19849303484
key up	132367	20097814864
key down	189008	20450483595
key up	143324	20719680135

Can We Build a Cache-Based Keylogger?



```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
```

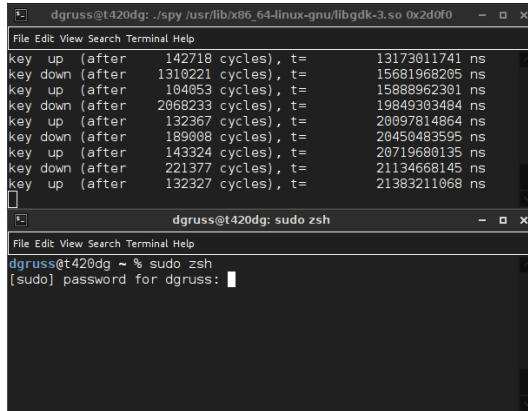
Event	Cycles	Time (ns)
key down	366253	12903870546
key up	142718	13173011741
key down	1310221	15681968205
key up	104053	15888962301
key down	2068233	19849303484
key up	132367	20097814864
key down	189008	20450483595
key up	143324	20719680135
key down	221377	21134668145

```
dgruss@t420dg: sudo zsh
```

```
dgruss@t420dg ~ % sudo zsh
```

```
[sudo] password for dgruss:
```

Can We Build a Cache-Based Keylogger?

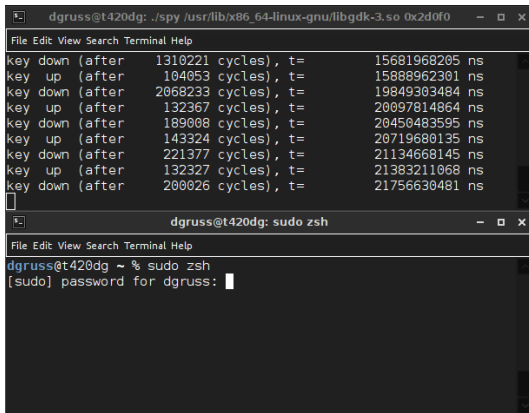


The image shows two terminal windows. The top window displays the output of a program running `./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0`. It shows a series of key events with timestamps in nanoseconds. The bottom window shows a `sudo zsh` prompt where the user's password is being requested.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 142718 cycles), t= 13173011741 ns
key down (after 1310221 cycles), t= 15681968205 ns
key up (after 104053 cycles), t= 15888962301 ns
key down (after 2068233 cycles), t= 19849303484 ns
key up (after 132367 cycles), t= 20097814864 ns
key down (after 189008 cycles), t= 20450483595 ns
key up (after 143324 cycles), t= 20719680135 ns
key down (after 221377 cycles), t= 21134668145 ns
key up (after 132327 cycles), t= 21383211068 ns

dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss: 
```

Can We Build a Cache-Based Keylogger?



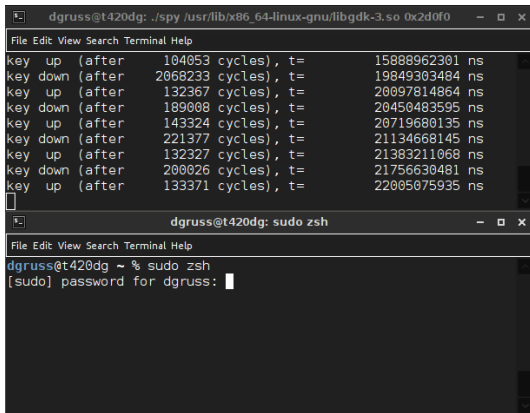
The image shows two terminal windows. The top window displays the output of a program running `./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0`. It shows a series of key events with timestamps in nanoseconds. The bottom window shows a `sudo zsh` prompt where the user's password is being requested.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key down (after 1310221 cycles), t= 15681968205 ns
key up (after 104053 cycles), t= 15888962301 ns
key down (after 2068233 cycles), t= 19849303484 ns
key up (after 132367 cycles), t= 20097814864 ns
key down (after 189008 cycles), t= 20450483595 ns
key up (after 143324 cycles), t= 20719680135 ns
key down (after 221377 cycles), t= 21134668145 ns
key up (after 132327 cycles), t= 21383211068 ns
key down (after 200026 cycles), t= 21756630481 ns

```

```
dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss: 
```

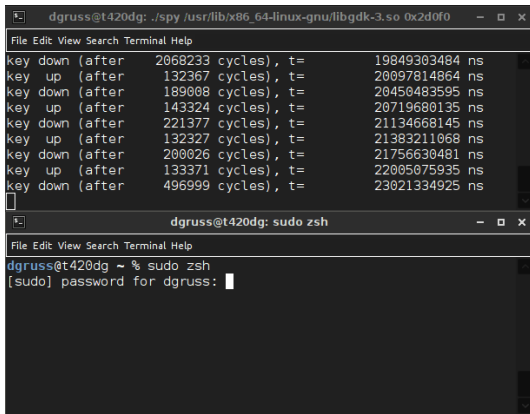
Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window, titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0', displays a list of key events with their cycle counts and timestamps. The bottom window, titled 'dgruss@t420dg: sudo zsh', shows the prompt '[sudo] password for dgruss:' with a cursor.

Event	Cycles	Time (ns)
key up (after	104053	15888962301
key down (after	2068233	19849303484
key up (after	132367	20097814864
key down (after	189008	20450483595
key up (after	143324	20719680135
key down (after	221377	21134668145
key up (after	132327	21383211068
key down (after	200026	21756630481
key up (after	133371	22005075935

Can We Build a Cache-Based Keylogger?

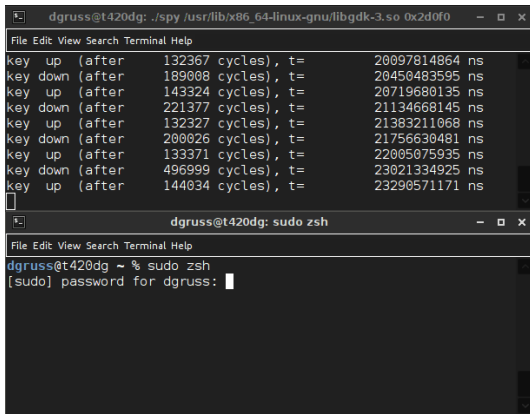


The image shows two terminal windows. The top window, titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0', displays a list of key events with their cycle counts and timestamps. The bottom window, titled 'dgruss@t420dg: sudo zsh', shows the prompt for the sudo password.

Event	Cycles	Timestamp (ns)
key down	2068233	19849303484
key up	132367	20097814864
key down	189008	20450483595
key up	143324	20719680135
key down	221377	21134668145
key up	132327	21383211068
key down	200026	21756630481
key up	133371	22005075935
key down	496999	23021334925

```
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
```

Can We Build a Cache-Based Keylogger?



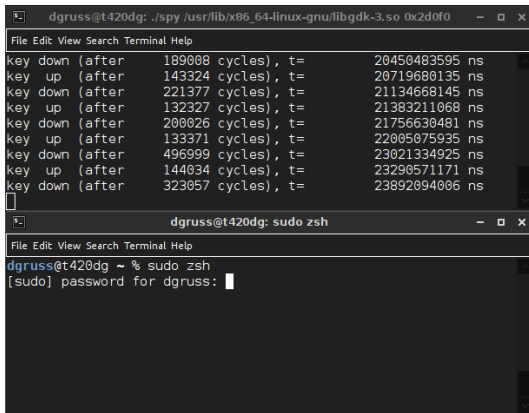
The image shows two terminal windows. The top window is titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and displays a list of key events with their cycle counts and timestamps. The bottom window is titled 'dgruss@t420dg: sudo zsh' and shows the user being prompted for their password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 132367 cycles), t= 20097814864 ns
key down (after 189008 cycles), t= 20450483595 ns
key up (after 143324 cycles), t= 20719680135 ns
key down (after 221377 cycles), t= 21134668145 ns
key up (after 132327 cycles), t= 21383211068 ns
key down (after 200026 cycles), t= 21756630481 ns
key up (after 133371 cycles), t= 22005075935 ns
key down (after 496999 cycles), t= 23021334925 ns
key up (after 144034 cycles), t= 23290571171 ns

```

```
dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss: 
```

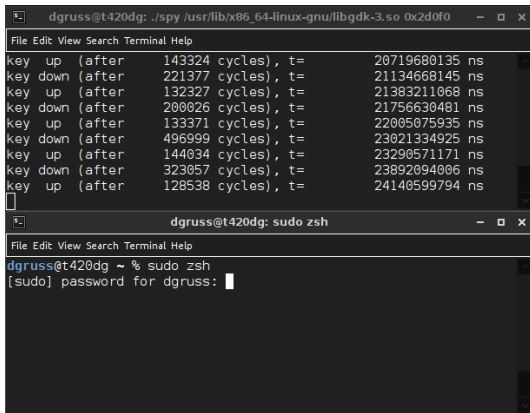
Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window is titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and displays a list of key events with their cycle counts and timestamps. The bottom window is titled 'dgruss@t420dg: sudo zsh' and shows the user being prompted for their password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key down (after 189008 cycles), t= 20450483595 ns
key up (after 143324 cycles), t= 20719680135 ns
key down (after 221377 cycles), t= 21134668145 ns
key up (after 132327 cycles), t= 21383211068 ns
key down (after 200026 cycles), t= 21756630481 ns
key up (after 133371 cycles), t= 22005075935 ns
key down (after 496999 cycles), t= 23021334925 ns
key up (after 144034 cycles), t= 23290571171 ns
key down (after 323057 cycles), t= 23892094006 ns
dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
```

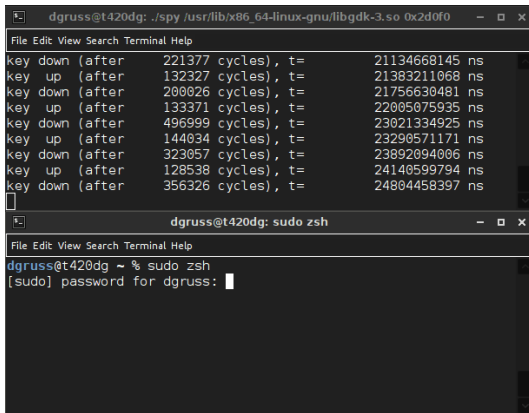
Can We Build a Cache-Based Keylogger?



```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 143324 cycles), t= 20719680135 ns
key down (after 221377 cycles), t= 21134668145 ns
key up (after 132327 cycles), t= 21383211068 ns
key down (after 200026 cycles), t= 21756630481 ns
key up (after 133371 cycles), t= 22005075935 ns
key down (after 496999 cycles), t= 23021334925 ns
key up (after 144034 cycles), t= 23290571171 ns
key down (after 323057 cycles), t= 23892094006 ns
key up (after 128538 cycles), t= 24140599794 ns

dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
```

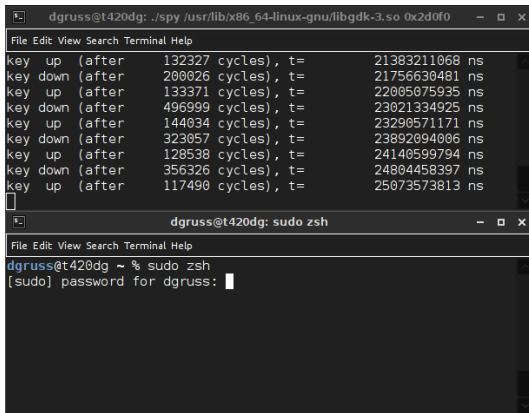
Can We Build a Cache-Based Keylogger?



```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key down (after 221377 cycles), t= 21134668145 ns
key up (after 132327 cycles), t= 21383211068 ns
key down (after 200026 cycles), t= 21756630481 ns
key up (after 133371 cycles), t= 22005075935 ns
key down (after 496999 cycles), t= 23021334925 ns
key up (after 144034 cycles), t= 23290571171 ns
key down (after 323057 cycles), t= 23892094006 ns
key up (after 128538 cycles), t= 24140599794 ns
key down (after 356326 cycles), t= 24804458397 ns

dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
```

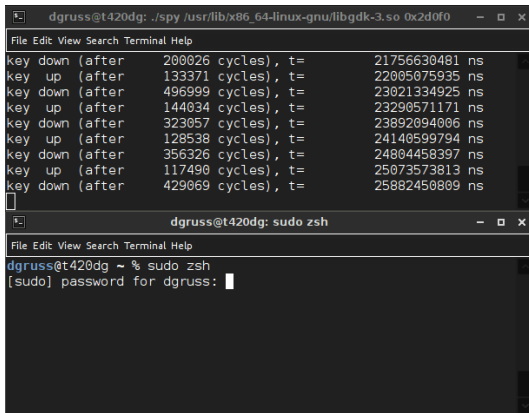
Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window, titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0', displays a list of key events with their cycle counts and timestamps. The bottom window, titled 'dgruss@t420dg: sudo zsh', shows the prompt '[sudo] password for dgruss:' with a cursor.

Event	Cycles	Timestamp (ns)
key up (after	132327	21383211068
key down (after	200026	21756630481
key up (after	133371	22005075935
key down (after	496999	23021334925
key up (after	144034	23290571171
key down (after	323057	23892094006
key up (after	128538	24140599794
key down (after	356326	24804458397
key up (after	117490	25073573813

Can We Build a Cache-Based Keylogger?

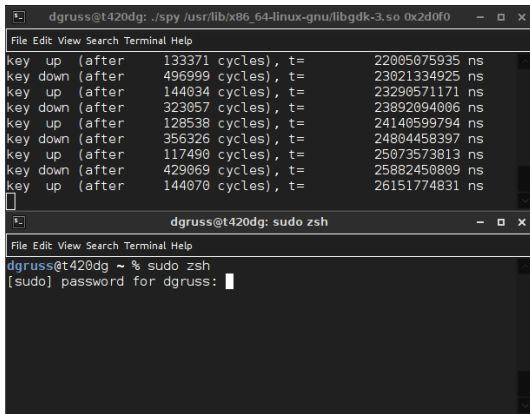


The image shows two terminal windows. The top window displays the output of a program running `./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0`. It shows a list of key events with their cycle counts and timestamps. The bottom window shows the user `dgruss` at `t420dg` running `sudo zsh`, which prompts for the password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key down (after 200026 cycles), t= 21756630481 ns
key up (after 133371 cycles), t= 22005075935 ns
key down (after 496999 cycles), t= 23021334925 ns
key up (after 144034 cycles), t= 23290571171 ns
key down (after 323057 cycles), t= 23892094006 ns
key up (after 128538 cycles), t= 24140599794 ns
key down (after 356326 cycles), t= 24804458397 ns
key up (after 117490 cycles), t= 25073573813 ns
key down (after 429069 cycles), t= 25882450809 ns
[ ]

dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss: [ ]
```

Can We Build a Cache-Based Keylogger?



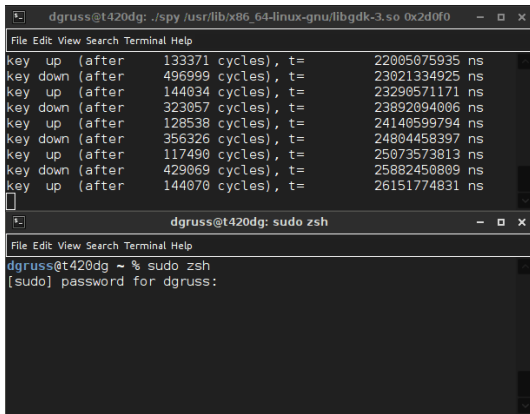
The image shows two terminal windows. The top window is titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and displays a list of key events with their cycle counts and timestamps. The bottom window is titled 'dgruss@t420dg: sudo zsh' and shows the prompt for the sudo password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 133371 cycles), t= 22005075935 ns
key down (after 496999 cycles), t= 23021334925 ns
key up (after 144034 cycles), t= 23290571171 ns
key down (after 323057 cycles), t= 23892094006 ns
key up (after 128538 cycles), t= 24140599794 ns
key down (after 356326 cycles), t= 24804458397 ns
key up (after 117490 cycles), t= 25073573813 ns
key down (after 429069 cycles), t= 25882450809 ns
key up (after 144070 cycles), t= 26151774831 ns

```

```
dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss: 
```

Can We Build a Cache-Based Keylogger?

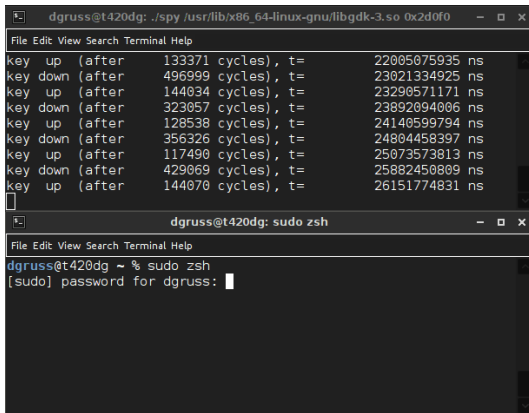


The image shows two terminal windows. The top window displays the output of a program running `./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0`. It shows a series of key events with timestamps in nanoseconds. The bottom window shows a shell prompt where the user has run `sudo zsh` and is prompted for a password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 133371 cycles), t= 22005075935 ns
key down (after 496999 cycles), t= 23021334925 ns
key up (after 144034 cycles), t= 23290571171 ns
key down (after 323057 cycles), t= 23892094006 ns
key up (after 128538 cycles), t= 24140599794 ns
key down (after 356326 cycles), t= 24804458397 ns
key up (after 117490 cycles), t= 25073573813 ns
key down (after 429069 cycles), t= 25882450809 ns
key up (after 144070 cycles), t= 26151774831 ns
[

dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
```

Can We Build a Cache-Based Keylogger?



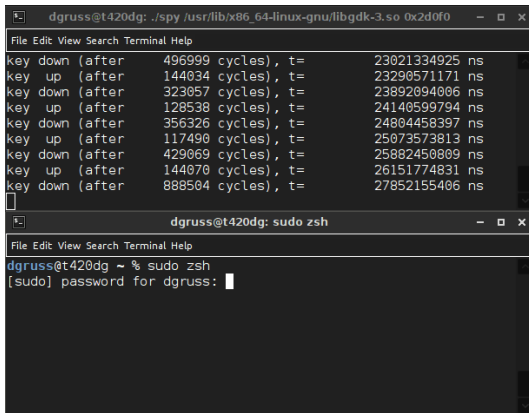
The image shows two terminal windows. The top window is titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and displays a list of key events with their cycle counts and timestamps. The bottom window is titled 'dgruss@t420dg: sudo zsh' and shows the prompt for the sudo password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 133371 cycles), t= 22005075935 ns
key down (after 496999 cycles), t= 23021334925 ns
key up (after 144034 cycles), t= 23290571171 ns
key down (after 323057 cycles), t= 23892094006 ns
key up (after 128538 cycles), t= 24140599794 ns
key down (after 356326 cycles), t= 24804458397 ns
key up (after 117490 cycles), t= 25073573813 ns
key down (after 429069 cycles), t= 25882450809 ns
key up (after 144070 cycles), t= 26151774831 ns

```

```
dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss: 
```

Can We Build a Cache-Based Keylogger?



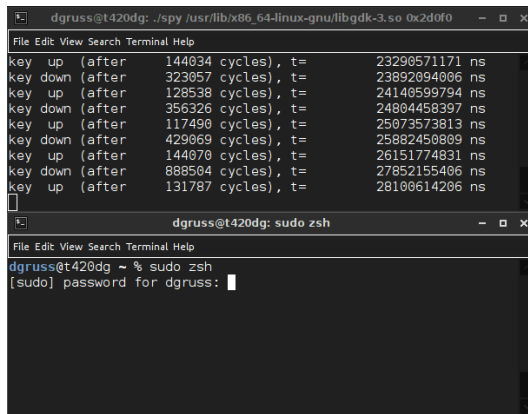
The image shows two terminal windows. The top window is titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and displays a list of key events with their cycle counts and timestamps. The bottom window is titled 'dgruss@t420dg: sudo zsh' and shows the user being prompted for their password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key down (after 496999 cycles), t= 23021334925 ns
key up (after 144034 cycles), t= 23290571171 ns
key down (after 323057 cycles), t= 23892094006 ns
key up (after 128538 cycles), t= 24140599794 ns
key down (after 356326 cycles), t= 24804458397 ns
key up (after 117490 cycles), t= 25073573813 ns
key down (after 429069 cycles), t= 25882450809 ns
key up (after 144070 cycles), t= 26151774831 ns
key down (after 888504 cycles), t= 27852155406 ns

```

```
dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss: 
```

Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window displays the output of a program that logs keypresses and the time taken for the CPU cache to refill. The data shows that keypresses are being logged and the time taken for the cache to refill is consistently around 23-28 nanoseconds. The bottom window shows a terminal prompt where the user has run 'sudo zsh' and is prompted for their password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
```

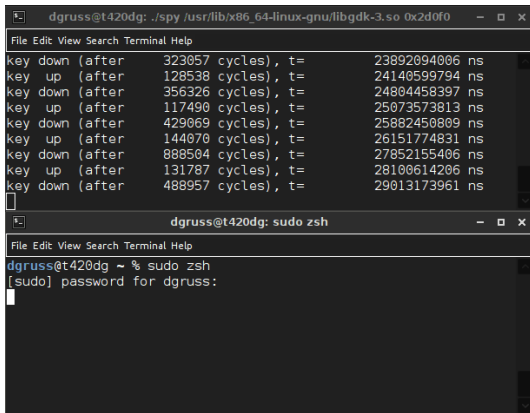
Key	Up/Down	Cycles	Time (ns)
key	up	(after 144034 cycles)	t= 23290571171 ns
key	down	(after 323057 cycles)	t= 23892094006 ns
key	up	(after 128538 cycles)	t= 24140599794 ns
key	down	(after 356326 cycles)	t= 24804458397 ns
key	up	(after 117490 cycles)	t= 25073573813 ns
key	down	(after 429069 cycles)	t= 25882450809 ns
key	up	(after 144070 cycles)	t= 26151774831 ns
key	down	(after 888504 cycles)	t= 27852155406 ns
key	up	(after 131787 cycles)	t= 28100614206 ns

```
dgruss@t420dg: sudo zsh
```

```
dgruss@t420dg ~ % sudo zsh
```

```
[sudo] password for dgruss:
```

Can We Build a Cache-Based Keylogger?

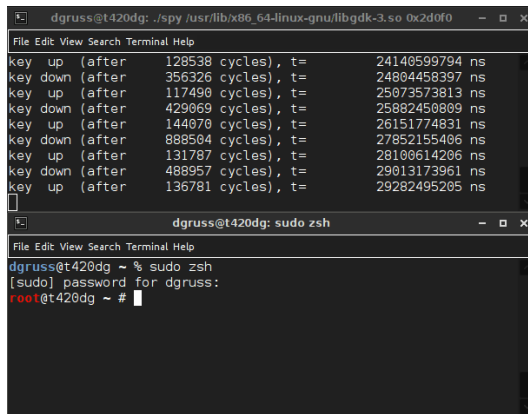


The image shows two terminal windows. The top window, titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0', displays a list of key events with their cycle counts and timestamps. The bottom window, titled 'dgruss@t420dg: sudo zsh', shows the prompt for the sudo password.

Event	Cycles	Timestamp (ns)
key down (after	323057	23892094006
key up (after	128538	24140599794
key down (after	356326	24804458397
key up (after	117490	25073573813
key down (after	429069	25882450809
key up (after	144070	26151774831
key down (after	888504	27852155406
key up (after	131787	28100614206
key down (after	488957	29013173961

```
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
```

Can We Build a Cache-Based Keylogger?

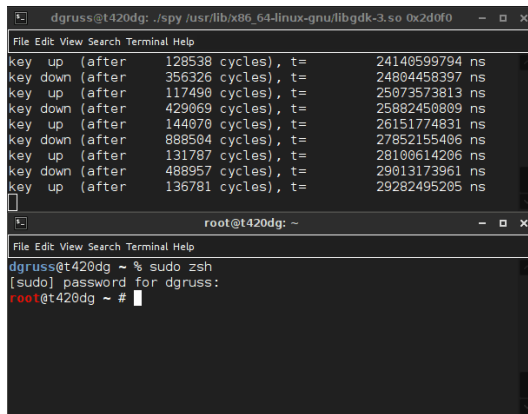


The image shows two terminal windows. The top window is titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0' and displays a list of key events with their cycle counts and timestamps. The bottom window is titled 'dgruss@t420dg: sudo zsh' and shows the user being prompted for a password and then becoming root.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 128538 cycles), t= 24140599794 ns
key down (after 356326 cycles), t= 24804458397 ns
key up (after 117490 cycles), t= 25073573813 ns
key down (after 429069 cycles), t= 25882450809 ns
key up (after 144070 cycles), t= 26151774831 ns
key down (after 888504 cycles), t= 27852155406 ns
key up (after 131787 cycles), t= 28100614206 ns
key down (after 488957 cycles), t= 29013173961 ns
key up (after 136781 cycles), t= 29282495205 ns

dgruss@t420dg: sudo zsh
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
root@t420dg ~ #
```

Can We Build a Cache-Based Keylogger?

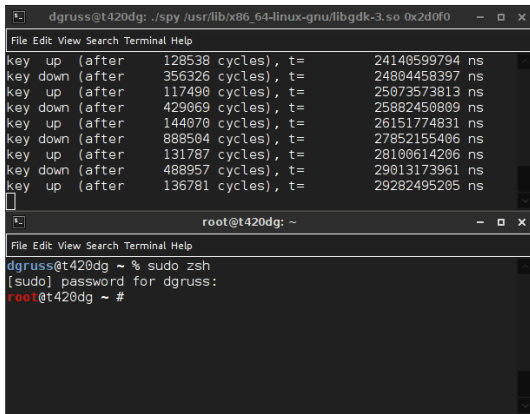


The image shows two terminal windows. The top window, titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0', displays a list of key events with their cycle counts and timestamps. The bottom window, titled 'root@t420dg: ~', shows the user 'dgruss' running 'sudo zsh' and being prompted for a password, after which they become the root user.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 128538 cycles), t= 24140599794 ns
key down (after 356326 cycles), t= 24804458397 ns
key up (after 117490 cycles), t= 25073573813 ns
key down (after 429069 cycles), t= 25882450809 ns
key up (after 144070 cycles), t= 26151774831 ns
key down (after 888504 cycles), t= 27852155406 ns
key up (after 131787 cycles), t= 28100614206 ns
key down (after 488957 cycles), t= 29013173961 ns
key up (after 136781 cycles), t= 29282495205 ns

root@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
root@t420dg ~ #
```

Can We Build a Cache-Based Keylogger?

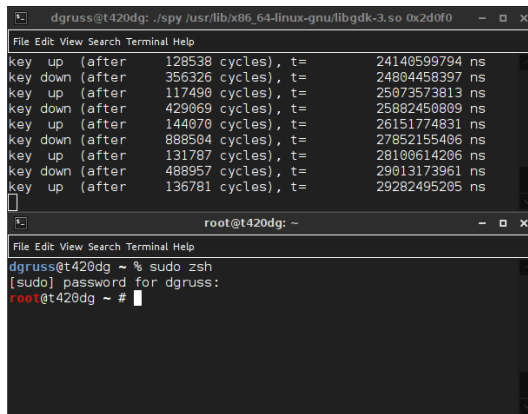


The image shows two terminal windows. The top window, titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0', displays a list of key events with their corresponding timestamps in nanoseconds. The bottom window, titled 'root@t420dg: ~', shows the user 'dgruss' using 'sudo zsh' to become root, with the prompt changing from '\$' to '#'.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 128538 cycles), t= 24140599794 ns
key down (after 356326 cycles), t= 24804458397 ns
key up (after 117490 cycles), t= 25073573813 ns
key down (after 429069 cycles), t= 25882450809 ns
key up (after 144070 cycles), t= 26151774831 ns
key down (after 888504 cycles), t= 27852155406 ns
key up (after 131787 cycles), t= 28100614206 ns
key down (after 488957 cycles), t= 29013173961 ns
key up (after 136781 cycles), t= 29282495205 ns

root@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
root@t420dg ~ #
```

Can We Build a Cache-Based Keylogger?

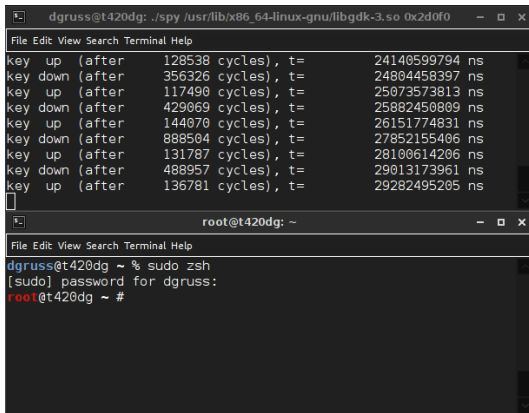


The image shows two terminal windows. The top window, titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0', displays a list of key events with their cycle counts and timestamps. The bottom window, titled 'root@t420dg: ~', shows the user 'dgruss' switching to a root shell using 'sudo zsh' and entering a password.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 128538 cycles), t= 24140599794 ns
key down (after 356326 cycles), t= 24804458397 ns
key up (after 117490 cycles), t= 25073573813 ns
key down (after 429069 cycles), t= 25882450809 ns
key up (after 144070 cycles), t= 26151774831 ns
key down (after 888504 cycles), t= 27852155406 ns
key up (after 131787 cycles), t= 28100614206 ns
key down (after 488957 cycles), t= 29013173961 ns
key up (after 136781 cycles), t= 29282495205 ns

root@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
root@t420dg ~ #
```

Can We Build a Cache-Based Keylogger?



The image shows two terminal windows. The top window, titled 'dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0', displays a list of key events with their corresponding timestamps in nanoseconds. The bottom window, titled 'root@t420dg: ~', shows the user 'dgruss' running 'sudo zsh' and being prompted for a password, after which they become the root user.

```
dgruss@t420dg: ./spy /usr/lib/x86_64-linux-gnu/libgdk-3.so 0x2d0f0
File Edit View Search Terminal Help
key up (after 128538 cycles), t= 24140599794 ns
key down (after 356326 cycles), t= 24804458397 ns
key up (after 117490 cycles), t= 25073573813 ns
key down (after 429069 cycles), t= 25882450809 ns
key up (after 144070 cycles), t= 26151774831 ns
key down (after 888504 cycles), t= 27852155406 ns
key up (after 131787 cycles), t= 28100614206 ns
key down (after 488957 cycles), t= 29013173961 ns
key up (after 136781 cycles), t= 29282495205 ns

root@t420dg: ~
File Edit View Search Terminal Help
dgruss@t420dg ~ % sudo zsh
[sudo] password for dgruss:
root@t420dg ~ #
```

Challenges

- How to locate key-dependent memory accesses?

Challenges

- How to locate key-dependent memory accesses?
- It's complicated:
 - Large binaries and libraries (third-party code)
 - Many libraries (gedit: 60MB)
 - Closed-source / unknown binaries
 - Self-compiled binaries

Challenges

- How to locate key-dependent memory accesses?
- It's complicated:
 - Large binaries and libraries (third-party code)
 - Many libraries (gedit: 60MB)
 - Closed-source / unknown binaries
 - Self-compiled binaries
- Difficult to find **all** exploitable addresses

Cache Template Attacks

Cache Template Attacks

Profiling Phase

- Preprocessing step to find exploitable addresses automatically
 - w.r.t. “events” (keystrokes, encryptions, ...)
 - called “Cache Template”

Cache Template Attacks

Profiling Phase

- Preprocessing step to find exploitable addresses automatically
 - w.r.t. “events” (keystrokes, encryptions, ...)
 - called “Cache Template”

Exploitation Phase

- Monitor exploitable addresses

Profiling Phase

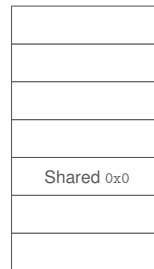
Attacker address space



Cache

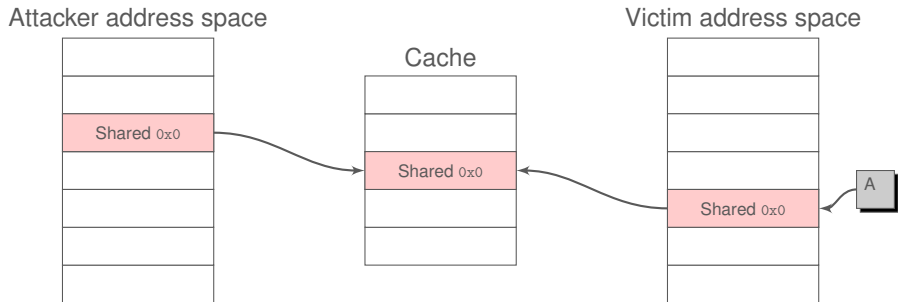


Victim address space



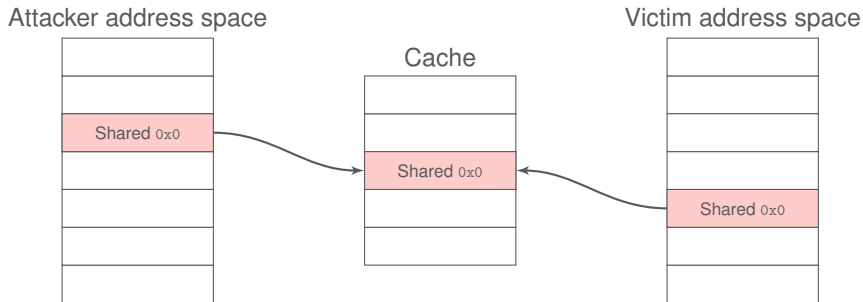
Cache is empty

Profiling Phase



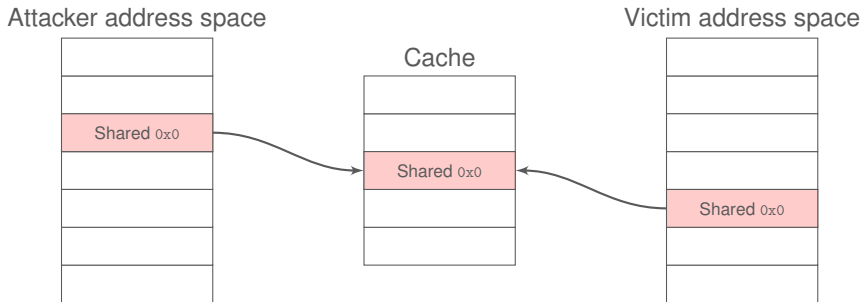
Attacker triggers an event

Profiling Phase



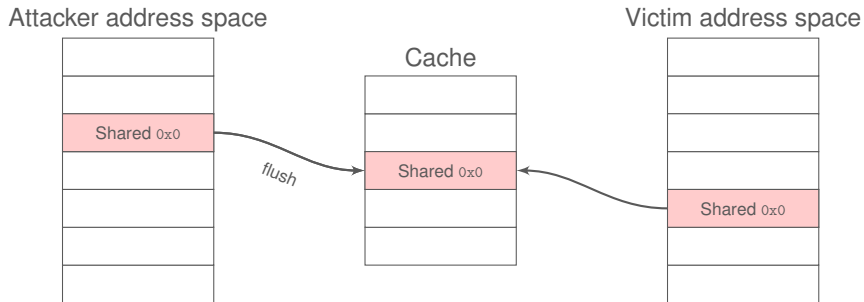
Attacker checks one address for cache hits (“Reload”)

Profiling Phase



Update cache hit ratio (per event and address)

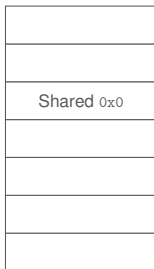
Profiling Phase



Attacker flushes shared memory

Profiling Phase

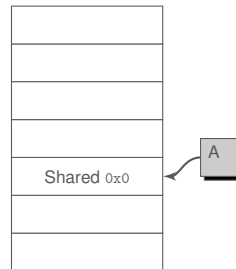
Attacker address space



Cache



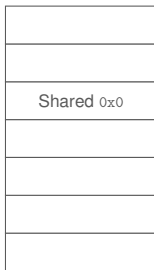
Victim address space



Repeat for higher accuracy

Profiling Phase

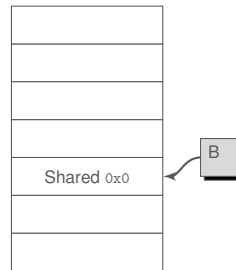
Attacker address space



Cache



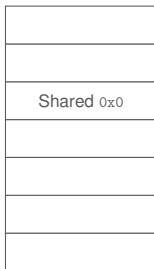
Victim address space



Repeat for all events

Profiling Phase

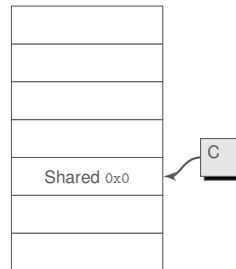
Attacker address space



Cache



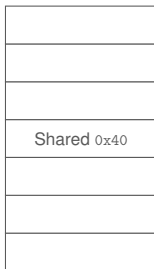
Victim address space



Repeat for all events

Profiling Phase

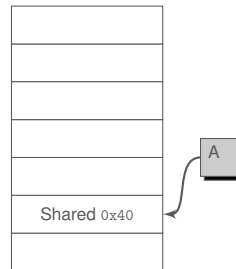
Attacker address space



Cache



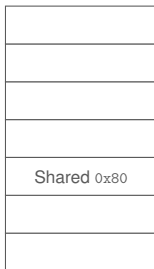
Victim address space



Continue with next address

Profiling Phase

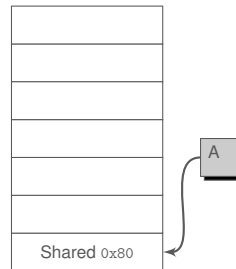
Attacker address space



Cache

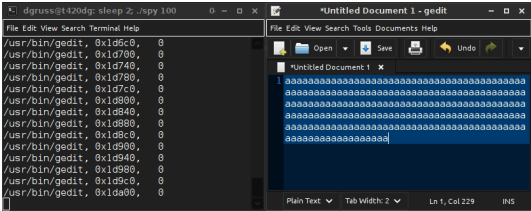


Victim address space

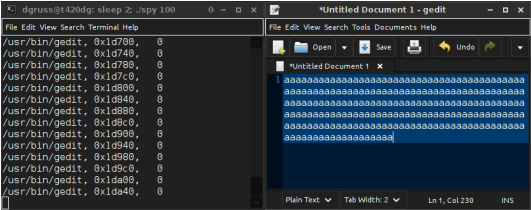


Continue with next address

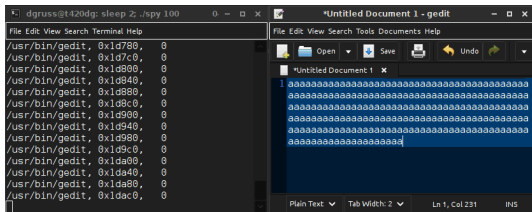
Profiling a Single Event



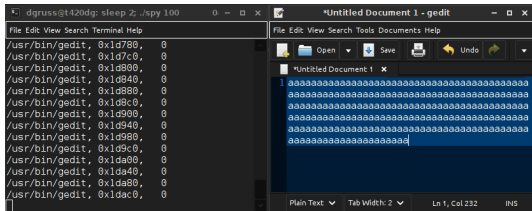
Profiling a Single Event



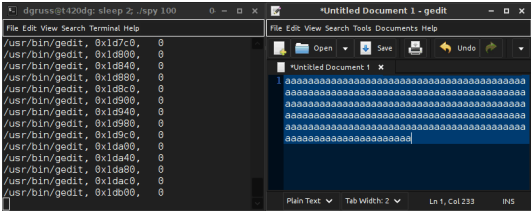
Profiling a Single Event



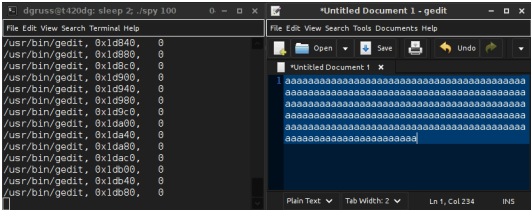
Profiling a Single Event



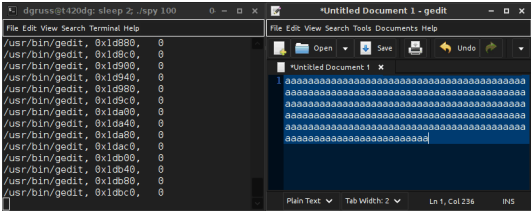
Profiling a Single Event



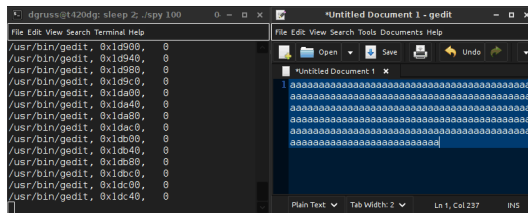
Profiling a Single Event



Profiling a Single Event



Profiling a Single Event

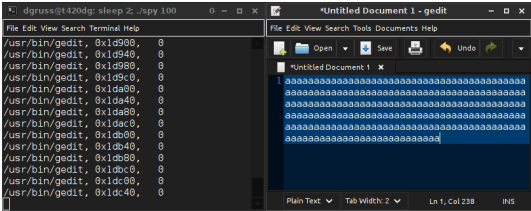


The screenshot shows two windows. The left window is a terminal with the prompt 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, all of which are '0'. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a memory dump with several lines of hexadecimal data, including addresses like 0x1d900, 0x1d940, 0x1d980, 0x1d9c0, 0x1da00, 0x1da40, 0x1da80, 0x1dac0, 0x1db00, 0x1db40, 0x1db80, 0x1dbc0, 0x1dc00, and 0x1dc40. The data values are represented by a series of 'a' characters.

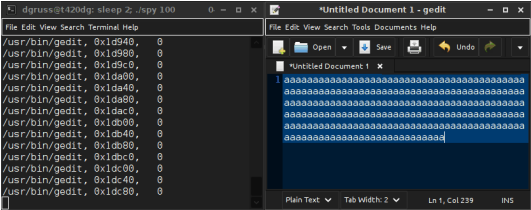
```
dgruss@t420dg: sleep 2; ./spy 100
File Edit View Search Terminal Help
/usr/bin/gedit, 0x1d900, 0
/usr/bin/gedit, 0x1d940, 0
/usr/bin/gedit, 0x1d980, 0
/usr/bin/gedit, 0x1d9c0, 0
/usr/bin/gedit, 0x1da00, 0
/usr/bin/gedit, 0x1da40, 0
/usr/bin/gedit, 0x1da80, 0
/usr/bin/gedit, 0x1dac0, 0
/usr/bin/gedit, 0x1db00, 0
/usr/bin/gedit, 0x1db40, 0
/usr/bin/gedit, 0x1db80, 0
/usr/bin/gedit, 0x1dbc0, 0
/usr/bin/gedit, 0x1dc00, 0
/usr/bin/gedit, 0x1dc40, 0

*Untitled Document 1 - gedit
File Edit View Search Tools Documents Help
Open Save Undo
*Untitled Document 1
| aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
Plain Text Tab Width: 2 Ln 1, Col 237 INS
```

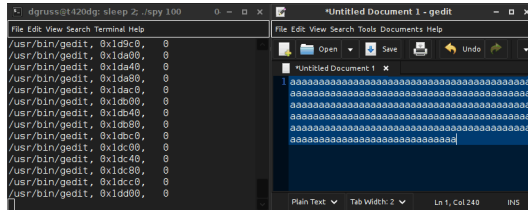
Profiling a Single Event



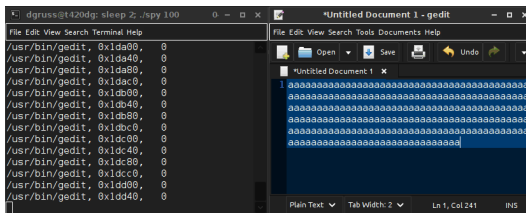
Profiling a Single Event



Profiling a Single Event



Profiling a Single Event

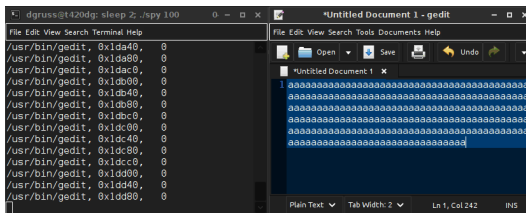


The screenshot shows two windows side-by-side. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, all of which are '0'. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a document with a large block of memory addresses (hexadecimal values) listed vertically, with the first few lines highlighted in blue. The status bar at the bottom of the gedit window indicates 'Ln 1, Col 241'.

```
dgruss@t420dg: sleep 2; ./spy 100
File Edit View Search Terminal Help
/usr/bin/gedit, 0x1da00, 0
/usr/bin/gedit, 0x1da40, 0
/usr/bin/gedit, 0x1da80, 0
/usr/bin/gedit, 0x1dac0, 0
/usr/bin/gedit, 0x1db00, 0
/usr/bin/gedit, 0x1db40, 0
/usr/bin/gedit, 0x1db80, 0
/usr/bin/gedit, 0x1dbc0, 0
/usr/bin/gedit, 0x1dc00, 0
/usr/bin/gedit, 0x1dc40, 0
/usr/bin/gedit, 0x1dc80, 0
/usr/bin/gedit, 0x1dcc0, 0
/usr/bin/gedit, 0x1dd00, 0
/usr/bin/gedit, 0x1dd40, 0

*Untitled Document 1 - gedit
File Edit View Search Tools Documents Help
Open Save Undo
*Untitled Document 1
1 aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
Plain Text Tab Width: 2 Ln 1, Col 241 INS
```

Profiling a Single Event

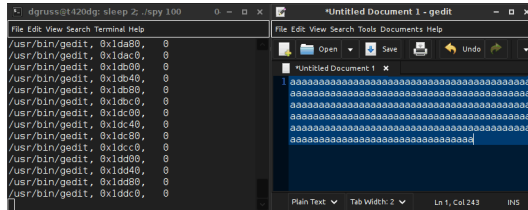


The screenshot shows two windows. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding hex values, all of which are '0'. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a document with several lines of hex data, which appear to be memory dumps or hex dumps. The data is organized into columns, with the first column containing addresses and the subsequent columns containing hex values. The gedit window also shows a menu bar with 'File', 'Edit', 'View', 'Search', 'Tools', 'Documents', and 'Help'. The status bar at the bottom of the gedit window indicates 'Plain Text', 'Tab Width: 2', 'Ln 1, Col 242', and 'INS'.

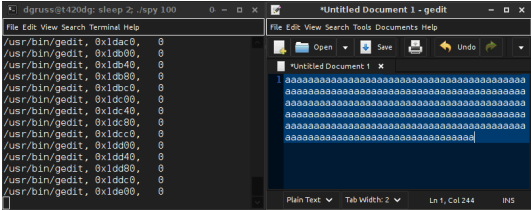
```
dgruss@t420dg: sleep 2; ./spy 100
File Edit View Search Terminal Help
/usr/bin/gedit, 0x1da40, 0
/usr/bin/gedit, 0x1da80, 0
/usr/bin/gedit, 0x1dac0, 0
/usr/bin/gedit, 0x1db00, 0
/usr/bin/gedit, 0x1db40, 0
/usr/bin/gedit, 0x1db80, 0
/usr/bin/gedit, 0x1dbc0, 0
/usr/bin/gedit, 0x1dc00, 0
/usr/bin/gedit, 0x1dc40, 0
/usr/bin/gedit, 0x1dc80, 0
/usr/bin/gedit, 0x1dcc0, 0
/usr/bin/gedit, 0x1dd00, 0
/usr/bin/gedit, 0x1dd40, 0
/usr/bin/gedit, 0x1dd80, 0

*Untitled Document 1 - gedit
File Edit View Search Tools Documents Help
*Untitled Document 1
| aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
Plain Text Tab Width: 2 Ln 1, Col 242 INS
```

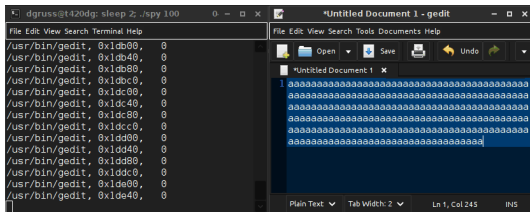
Profiling a Single Event



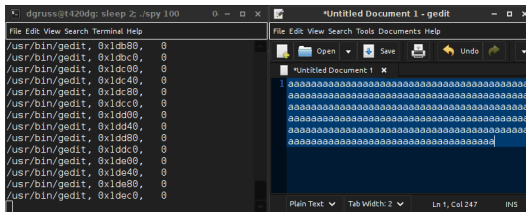
Profiling a Single Event



Profiling a Single Event



Profiling a Single Event

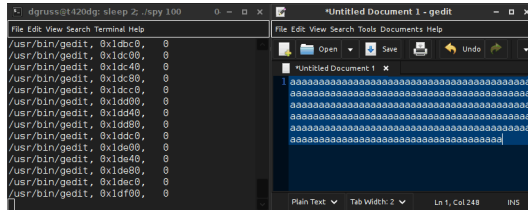


The screenshot shows two windows. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, all of which are 0. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a document with a single line of memory addresses, all of which are 0.

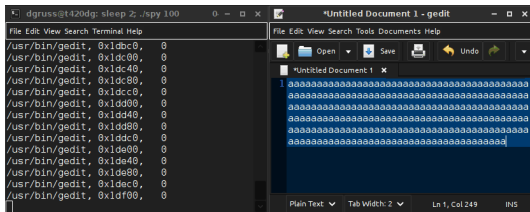
```
dgruss@t420dg: sleep 2; ./spy 100
/usr/bin/gedit, 0x1db80, 0
/usr/bin/gedit, 0x1dbc0, 0
/usr/bin/gedit, 0x1dc00, 0
/usr/bin/gedit, 0x1dc40, 0
/usr/bin/gedit, 0x1dc80, 0
/usr/bin/gedit, 0x1dcc0, 0
/usr/bin/gedit, 0x1dd00, 0
/usr/bin/gedit, 0x1dd40, 0
/usr/bin/gedit, 0x1dd80, 0
/usr/bin/gedit, 0x1ddc0, 0
/usr/bin/gedit, 0x1de00, 0
/usr/bin/gedit, 0x1de40, 0
/usr/bin/gedit, 0x1de80, 0
/usr/bin/gedit, 0x1dec0, 0
```

```
*Untitled Document 1 - gedit
0x1db80, 0
0x1dbc0, 0
0x1dc00, 0
0x1dc40, 0
0x1dc80, 0
0x1dcc0, 0
0x1dd00, 0
0x1dd40, 0
0x1dd80, 0
0x1ddc0, 0
0x1de00, 0
0x1de40, 0
0x1de80, 0
0x1dec0, 0
```

Profiling a Single Event



Profiling a Single Event

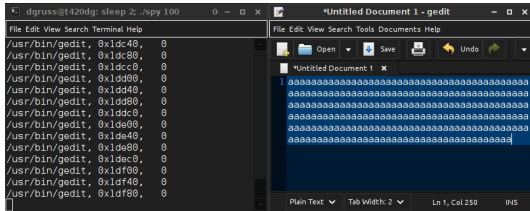


The screenshot shows two windows side-by-side. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, all of which are 0. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a document with several lines of memory addresses, each followed by a 0. The addresses are: 0x1dbc0, 0x1dc00, 0x1dc40, 0x1dc80, 0x1dcc0, 0x1dd00, 0x1dd40, 0x1dd80, 0x1ddc0, 0x1de00, 0x1de40, 0x1de80, 0x1dec0, and 0x1df00.

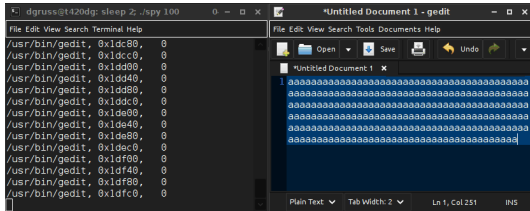
```
dgruss@t420dg: sleep 2; ./spy 100
/usr/bin/gedit, 0x1dbc0, 0
/usr/bin/gedit, 0x1dc00, 0
/usr/bin/gedit, 0x1dc40, 0
/usr/bin/gedit, 0x1dc80, 0
/usr/bin/gedit, 0x1dcc0, 0
/usr/bin/gedit, 0x1dd00, 0
/usr/bin/gedit, 0x1dd40, 0
/usr/bin/gedit, 0x1dd80, 0
/usr/bin/gedit, 0x1ddc0, 0
/usr/bin/gedit, 0x1de00, 0
/usr/bin/gedit, 0x1de40, 0
/usr/bin/gedit, 0x1de80, 0
/usr/bin/gedit, 0x1dec0, 0
/usr/bin/gedit, 0x1df00, 0
```

```
*Untitled Document 1 - gedit
File Edit View Search Tools Documents Help
Open Save Undo
*Untitled Document 1
0x1dbc0, 0
0x1dc00, 0
0x1dc40, 0
0x1dc80, 0
0x1dcc0, 0
0x1dd00, 0
0x1dd40, 0
0x1dd80, 0
0x1ddc0, 0
0x1de00, 0
0x1de40, 0
0x1de80, 0
0x1dec0, 0
0x1df00, 0
```

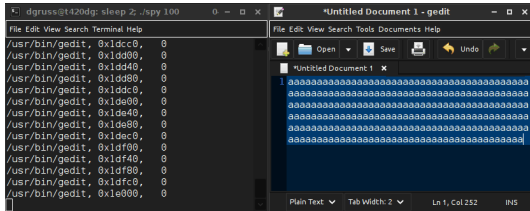
Profiling a Single Event



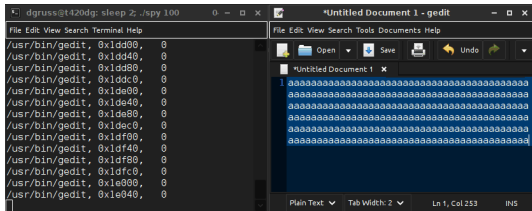
Profiling a Single Event



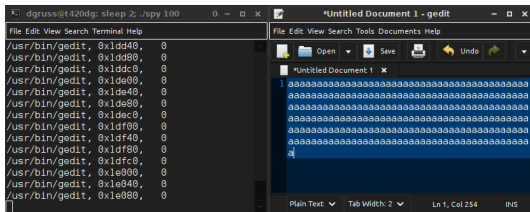
Profiling a Single Event



Profiling a Single Event



Profiling a Single Event

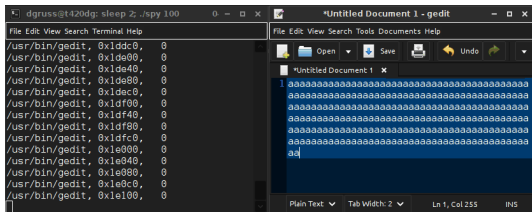


The screenshot shows two windows side-by-side. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding hex values, all of which are '0'. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a document with a single line of text consisting of a long sequence of 'a' characters, followed by a single '0' character on the next line. The status bar at the bottom of the gedit window indicates 'Ln 1, Col 254'.

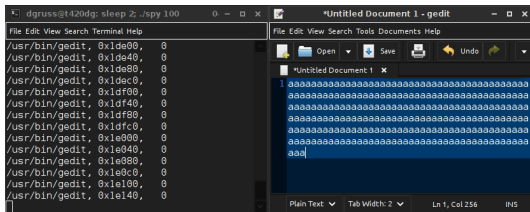
```
dgruss@t420dg: sleep 2; ./spy 100
File Edit View Search Terminal Help
/usr/bin/gedit, 0x1dd40, 0
/usr/bin/gedit, 0x1dd80, 0
/usr/bin/gedit, 0x1ddc0, 0
/usr/bin/gedit, 0x1de00, 0
/usr/bin/gedit, 0x1de40, 0
/usr/bin/gedit, 0x1de80, 0
/usr/bin/gedit, 0x1dec0, 0
/usr/bin/gedit, 0x1df00, 0
/usr/bin/gedit, 0x1df40, 0
/usr/bin/gedit, 0x1df80, 0
/usr/bin/gedit, 0x1dfc0, 0
/usr/bin/gedit, 0x1e000, 0
/usr/bin/gedit, 0x1e040, 0
/usr/bin/gedit, 0x1e080, 0

*Untitled Document 1 - gedit
File Edit View Search Tools Documents Help
Open Save Undo
*Untitled Document 1
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
a
Plain Text Tab Width: 2 Ln 1, Col 254 INS
```

Profiling a Single Event



Profiling a Single Event

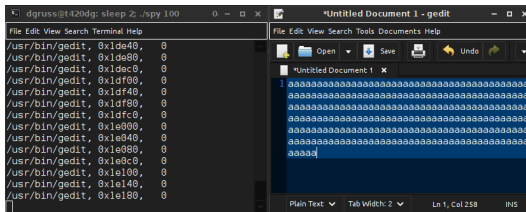


The screenshot shows two windows side-by-side. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, all of which are '0'. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a large block of 'a' characters, with the cursor positioned at the end of the first line. The status bar at the bottom of the gedit window indicates 'Ln 1, Col 256'.

```
dgruss@t420dg: sleep 2; ./spy 100
/usr/bin/gedit, 0x1de00, 0
/usr/bin/gedit, 0x1de40, 0
/usr/bin/gedit, 0x1de80, 0
/usr/bin/gedit, 0x1dec0, 0
/usr/bin/gedit, 0x1df00, 0
/usr/bin/gedit, 0x1df40, 0
/usr/bin/gedit, 0x1df80, 0
/usr/bin/gedit, 0x1dfc0, 0
/usr/bin/gedit, 0x1e000, 0
/usr/bin/gedit, 0x1e040, 0
/usr/bin/gedit, 0x1e080, 0
/usr/bin/gedit, 0x1e0c0, 0
/usr/bin/gedit, 0x1e100, 0
/usr/bin/gedit, 0x1e140, 0
```

```
*Untitled Document 1 - gedit
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaa
```

Profiling a Single Event

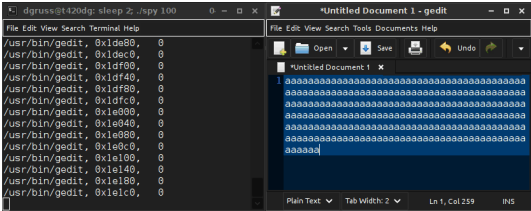


The screenshot shows two windows. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, all of which are 0. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a memory dump with several lines of hexadecimal data, followed by a line of 'aaaaa'.

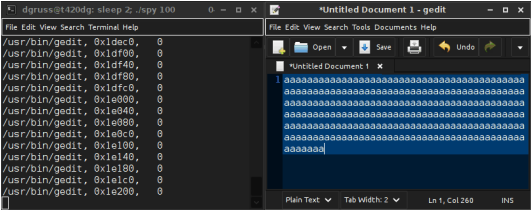
```
dgruss@t420dg: sleep 2; ./spy 100
/usr/bin/gedit, 0x1de40, 0
/usr/bin/gedit, 0x1de80, 0
/usr/bin/gedit, 0x1dec0, 0
/usr/bin/gedit, 0x1df00, 0
/usr/bin/gedit, 0x1df40, 0
/usr/bin/gedit, 0x1df80, 0
/usr/bin/gedit, 0x1dfc0, 0
/usr/bin/gedit, 0x1e000, 0
/usr/bin/gedit, 0x1e040, 0
/usr/bin/gedit, 0x1e080, 0
/usr/bin/gedit, 0x1e0c0, 0
/usr/bin/gedit, 0x1e100, 0
/usr/bin/gedit, 0x1e140, 0
/usr/bin/gedit, 0x1e180, 0

*Untitled Document 1 - gedit
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaa
```

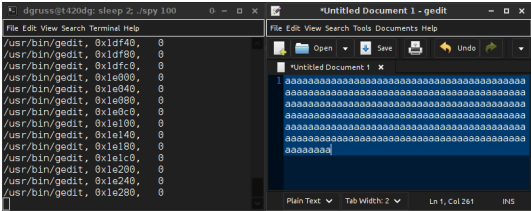
Profiling a Single Event



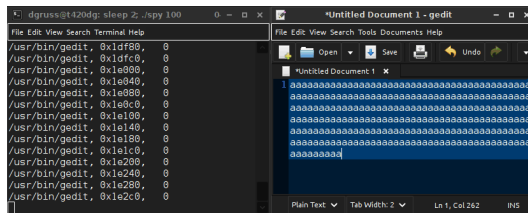
Profiling a Single Event



Profiling a Single Event



Profiling a Single Event

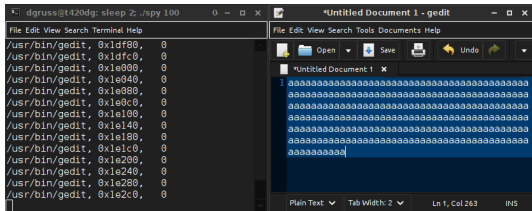


The screenshot shows two windows side-by-side. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding hex values, all starting with '/usr/bin/gedit,'. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a document with multiple lines of hex data, with the first line highlighted in blue. The status bar at the bottom of the gedit window indicates 'Plain Text', 'Tab Width: 2', 'Ln 1, Col 262', and 'INS'.

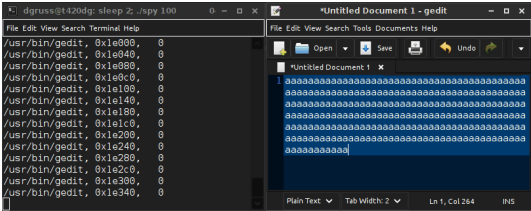
```
dgruss@t420dg: sleep 2; ./spy 100
File Edit View Search Terminal Help
/usr/bin/gedit, 0x1df80, 0
/usr/bin/gedit, 0x1dfc0, 0
/usr/bin/gedit, 0x1e000, 0
/usr/bin/gedit, 0x1e040, 0
/usr/bin/gedit, 0x1e080, 0
/usr/bin/gedit, 0x1e0c0, 0
/usr/bin/gedit, 0x1e100, 0
/usr/bin/gedit, 0x1e140, 0
/usr/bin/gedit, 0x1e180, 0
/usr/bin/gedit, 0x1e1c0, 0
/usr/bin/gedit, 0x1e200, 0
/usr/bin/gedit, 0x1e240, 0
/usr/bin/gedit, 0x1e280, 0
/usr/bin/gedit, 0x1e2c0, 0

*Untitled Document 1 - gedit
File Edit View Search Tools Documents Help
Open Save Undo
*Untitled Document 1
1 aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaa
Plain Text Tab Width: 2 Ln 1, Col 262 INS
```

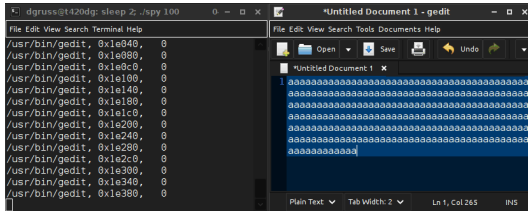
Profiling a Single Event



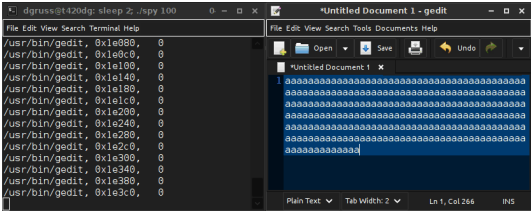
Profiling a Single Event



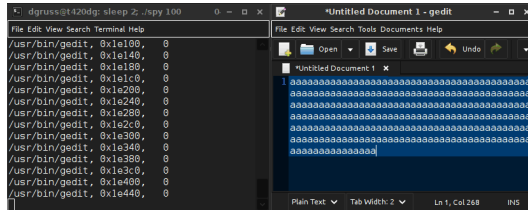
Profiling a Single Event



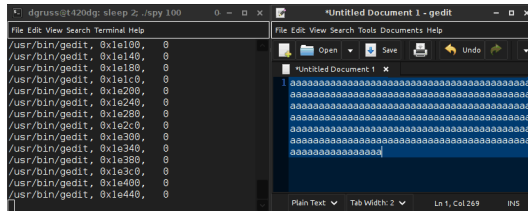
Profiling a Single Event



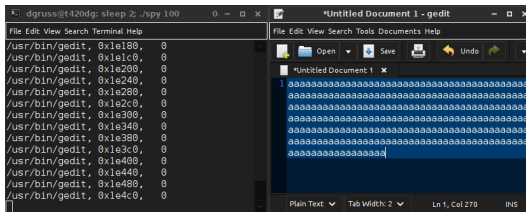
Profiling a Single Event



Profiling a Single Event

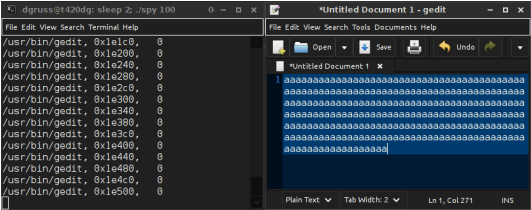


Profiling a Single Event

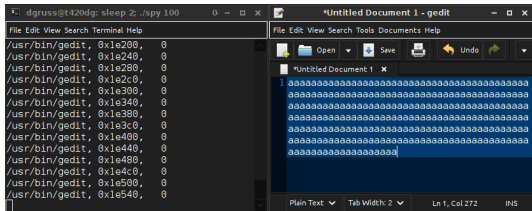


The screenshot shows two windows. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, all of which are 0. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a memory dump with several lines of hexadecimal data, including addresses like 0x1e180, 0x1e1c0, 0x1e200, 0x1e240, 0x1e280, 0x1e2c0, 0x1e300, 0x1e340, 0x1e380, 0x1e3c0, 0x1e400, 0x1e440, 0x1e480, and 0x1e4c0. The values are mostly 0, with some lines showing a large block of 'a' characters, likely representing a memory dump or a specific event being profiled.

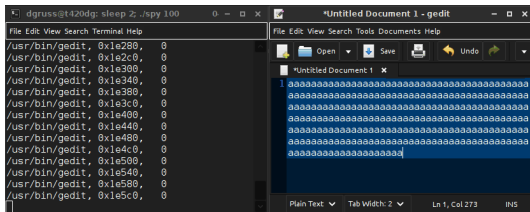
Profiling a Single Event



Profiling a Single Event



Profiling a Single Event

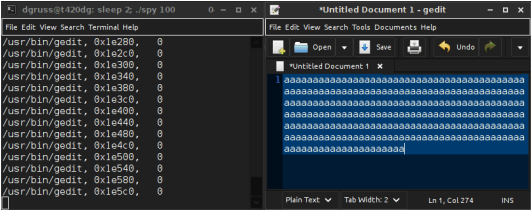


The screenshot shows two windows side-by-side. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, all of which are '0'. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a large block of 'a' characters, indicating a memory dump or a specific event being profiled. The status bar at the bottom of the gedit window shows 'Ln 1, Col 273' and 'INS'.

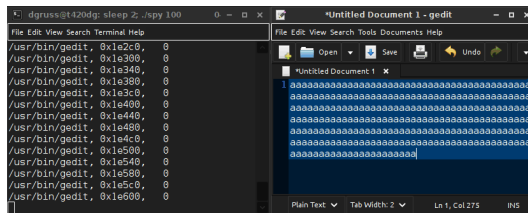
```
dgruss@t420dg: sleep 2; ./spy 100
/usr/bin/gedit, 0x1e280, 0
/usr/bin/gedit, 0x1e2c0, 0
/usr/bin/gedit, 0x1e300, 0
/usr/bin/gedit, 0x1e340, 0
/usr/bin/gedit, 0x1e380, 0
/usr/bin/gedit, 0x1e3c0, 0
/usr/bin/gedit, 0x1e400, 0
/usr/bin/gedit, 0x1e440, 0
/usr/bin/gedit, 0x1e480, 0
/usr/bin/gedit, 0x1e4c0, 0
/usr/bin/gedit, 0x1e500, 0
/usr/bin/gedit, 0x1e540, 0
/usr/bin/gedit, 0x1e580, 0
/usr/bin/gedit, 0x1e5c0, 0
```

```
*Untitled Document 1 - gedit
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
```

Profiling a Single Event



Profiling a Single Event

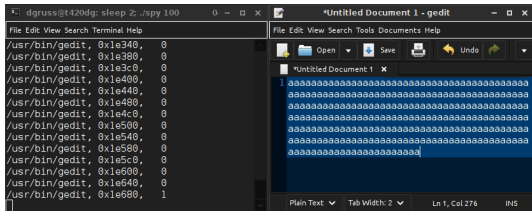


The screenshot shows two windows side-by-side. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, all of which are 0. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a large block of memory dump, consisting of many lines of hexadecimal characters (a's and b's) representing memory data.

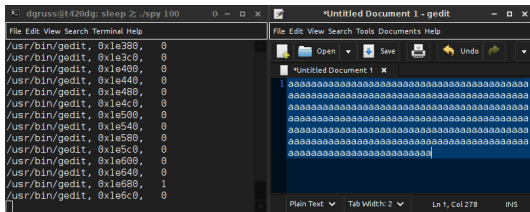
```
dgruss@t420dg: sleep 2; ./spy 100
/usr/bin/gedit, 0x1e2c0, 0
/usr/bin/gedit, 0x1e300, 0
/usr/bin/gedit, 0x1e340, 0
/usr/bin/gedit, 0x1e380, 0
/usr/bin/gedit, 0x1e3c0, 0
/usr/bin/gedit, 0x1e400, 0
/usr/bin/gedit, 0x1e440, 0
/usr/bin/gedit, 0x1e480, 0
/usr/bin/gedit, 0x1e4c0, 0
/usr/bin/gedit, 0x1e500, 0
/usr/bin/gedit, 0x1e540, 0
/usr/bin/gedit, 0x1e580, 0
/usr/bin/gedit, 0x1e5c0, 0
/usr/bin/gedit, 0x1e600, 0
```

```
*Untitled Document 1 - gedit
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
```

Profiling a Single Event



Profiling a Single Event

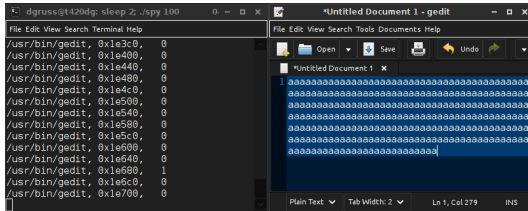


The screenshot shows two windows side-by-side. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and values, all starting with '/usr/bin/gedit,':

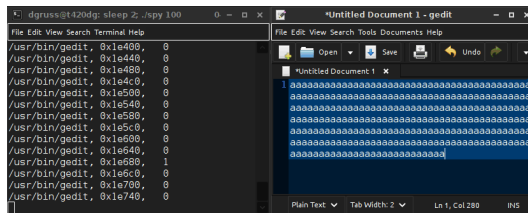
```
/usr/bin/gedit, 0x1e380, 0
/usr/bin/gedit, 0x1e3c0, 0
/usr/bin/gedit, 0x1e400, 0
/usr/bin/gedit, 0x1e440, 0
/usr/bin/gedit, 0x1e480, 0
/usr/bin/gedit, 0x1e4c0, 0
/usr/bin/gedit, 0x1e500, 0
/usr/bin/gedit, 0x1e540, 0
/usr/bin/gedit, 0x1e580, 0
/usr/bin/gedit, 0x1e5c0, 0
/usr/bin/gedit, 0x1e600, 0
/usr/bin/gedit, 0x1e640, 0
/usr/bin/gedit, 0x1e680, 1
/usr/bin/gedit, 0x1e6c0, 0
```

The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a large block of text consisting of many lines of 'a' characters, indicating a memory dump or a large string of data.

Profiling a Single Event



Profiling a Single Event

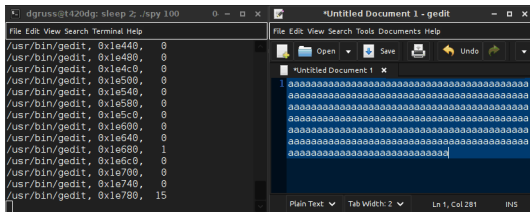


The screenshot shows two windows side-by-side. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, all in hexadecimal. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a document with a single line of text consisting of a long sequence of 'a' characters, which is highlighted in blue. The terminal output is as follows:

```
dgruss@t420dg: sleep 2; ./spy 100
/usr/bin/gedit, 0x1e400, 0
/usr/bin/gedit, 0x1e440, 0
/usr/bin/gedit, 0x1e480, 0
/usr/bin/gedit, 0x1e4c0, 0
/usr/bin/gedit, 0x1e500, 0
/usr/bin/gedit, 0x1e540, 0
/usr/bin/gedit, 0x1e580, 0
/usr/bin/gedit, 0x1e5c0, 0
/usr/bin/gedit, 0x1e600, 0
/usr/bin/gedit, 0x1e640, 0
/usr/bin/gedit, 0x1e680, 1
/usr/bin/gedit, 0x1e6c0, 0
/usr/bin/gedit, 0x1e700, 0
/usr/bin/gedit, 0x1e740, 0
```

The gedit editor shows a document with a single line of text consisting of a long sequence of 'a' characters, which is highlighted in blue.

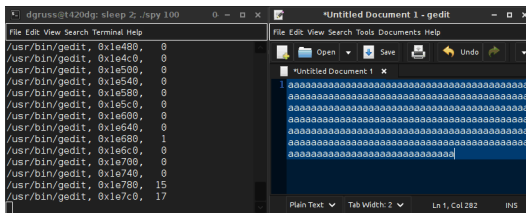
Profiling a Single Event



The screenshot shows two windows side-by-side. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, representing a stack trace or profiling data. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a document with a single line of text consisting of a long string of 'a' characters, which is highlighted in blue. The terminal output is as follows:

```
dgruss@t420dg: sleep 2; ./spy 100
/usr/bin/gedit, 0x1e440, 0
/usr/bin/gedit, 0x1e480, 0
/usr/bin/gedit, 0x1e4c0, 0
/usr/bin/gedit, 0x1e500, 0
/usr/bin/gedit, 0x1e540, 0
/usr/bin/gedit, 0x1e580, 0
/usr/bin/gedit, 0x1e5c0, 0
/usr/bin/gedit, 0x1e600, 0
/usr/bin/gedit, 0x1e640, 0
/usr/bin/gedit, 0x1e680, 1
/usr/bin/gedit, 0x1e6c0, 0
/usr/bin/gedit, 0x1e700, 0
/usr/bin/gedit, 0x1e740, 0
/usr/bin/gedit, 0x1e780, 15
```

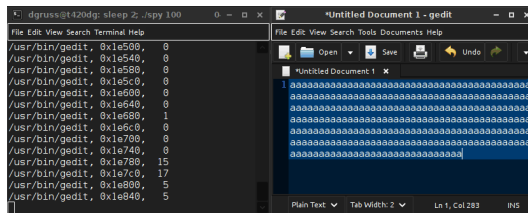
Profiling a Single Event



The screenshot shows two windows. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, likely from a memory dump or profiler output. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a document with a single line of text consisting of a long string of 'a' characters, which is highlighted in blue. The terminal output is as follows:

```
dgruss@t420dg: sleep 2; ./spy 100
/usr/bin/gedit, 0x1e480, 0
/usr/bin/gedit, 0x1e4c0, 0
/usr/bin/gedit, 0x1e500, 0
/usr/bin/gedit, 0x1e540, 0
/usr/bin/gedit, 0x1e580, 0
/usr/bin/gedit, 0x1e5c0, 0
/usr/bin/gedit, 0x1e600, 0
/usr/bin/gedit, 0x1e640, 0
/usr/bin/gedit, 0x1e680, 1
/usr/bin/gedit, 0x1e6c0, 0
/usr/bin/gedit, 0x1e700, 0
/usr/bin/gedit, 0x1e740, 0
/usr/bin/gedit, 0x1e780, 15
/usr/bin/gedit, 0x1e7c0, 17
```

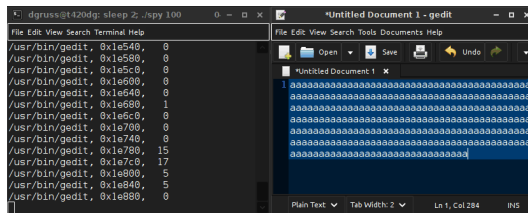
Profiling a Single Event



The screenshot shows two windows. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, representing profiling data. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a document with a large block of memory addresses, likely the same data as the terminal, but with some lines highlighted in blue.

```
dgruss@t420dg: sleep 2; ./spy 100
/usr/bin/gedit, 0x1e500, 0
/usr/bin/gedit, 0x1e540, 0
/usr/bin/gedit, 0x1e580, 0
/usr/bin/gedit, 0x1e5c0, 0
/usr/bin/gedit, 0x1e600, 0
/usr/bin/gedit, 0x1e640, 0
/usr/bin/gedit, 0x1e680, 1
/usr/bin/gedit, 0x1e6c0, 0
/usr/bin/gedit, 0x1e700, 0
/usr/bin/gedit, 0x1e740, 0
/usr/bin/gedit, 0x1e780, 15
/usr/bin/gedit, 0x1e7c0, 17
/usr/bin/gedit, 0x1e800, 5
/usr/bin/gedit, 0x1e840, 5
```

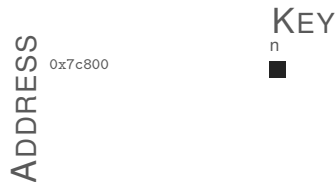
Profiling a Single Event



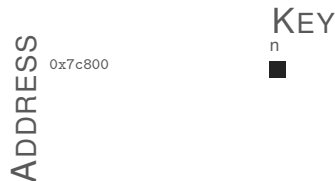
The screenshot shows two windows. The left window is a terminal titled 'dgruss@t420dg: sleep 2; ./spy 100'. It displays a list of memory addresses and their corresponding values, likely from a memory dump or profiler output. The right window is a gedit editor titled '*Untitled Document 1 - gedit'. It shows a document with a large block of text consisting of many lines of 'a' characters, which appears to be a memory dump or a log of data. The status bar at the bottom of the gedit window indicates 'Ln 1, Col 284' and 'INS'.

```
dgruss@t420dg: sleep 2; ./spy 100
/usr/bin/gedit, 0x1e540, 0
/usr/bin/gedit, 0x1e580, 0
/usr/bin/gedit, 0x1e5c0, 0
/usr/bin/gedit, 0x1e600, 0
/usr/bin/gedit, 0x1e640, 0
/usr/bin/gedit, 0x1e680, 1
/usr/bin/gedit, 0x1e6c0, 0
/usr/bin/gedit, 0x1e700, 0
/usr/bin/gedit, 0x1e740, 0
/usr/bin/gedit, 0x1e780, 15
/usr/bin/gedit, 0x1e7c0, 17
/usr/bin/gedit, 0x1e800, 5
/usr/bin/gedit, 0x1e840, 5
/usr/bin/gedit, 0x1e880, 0
```

Profiling Phase: 1 Event, 1 Address



Profiling Phase: 1 Event, 1 Address



Example: Cache Hit Ratio for $(0x7c800, n)$: 200 / 200

Profiling Phase: All Events, 1 Address



Profiling Phase: All Events, 1 Address



Example: Cache Hit Ratio for $(0x7c800, u)$: 13 / 200

Profiling Phase: All Events, 1 Address



Distinguish n from other keys by monitoring 0x7c800

Profiling Phase: All Events, All Addresses



Exploitation Phase

- Monitor addresses from Cache Template

Exploitation Phase

- Monitor addresses from Cache Template
- Report to log file / attacker

Exploitation Phase

- Monitor addresses from Cache Template
- Report to log file / attacker
- Manual analysis of log file
 - Find password in keypress log, etc.

Example Attacks

Attack 1: Keystroke Timings

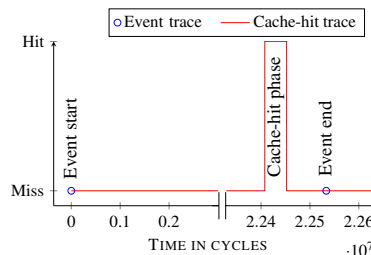
- Spy on keystroke timings on Linux, Windows and OS X

Attack 1: Keystroke Timings

- Spy on keystroke timings on Linux, Windows and OS X
- Sub-microsecond accuracy

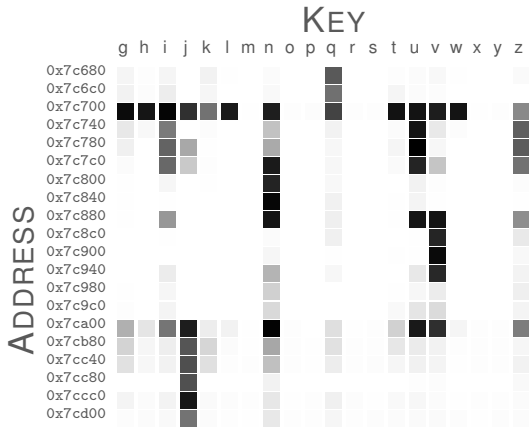
Attack 1: Keystroke Timings

- Spy on keystroke timings on Linux, Windows and OS X
- Sub-microsecond accuracy
- Derive text input from timings



Attack 2: Keylogging

- Linux with GTK: monitor keystrokes of specific keys
- Detect groups of keys
- Some keys distinct



Attack 3: Locate AES T-Tables

AES T-Table implementation from OpenSSL 1.0.2

Attack 3: Locate AES T-Tables

AES T-Table implementation from OpenSSL 1.0.2

- Most addresses in two groups:
 - Cache hit ratio 100% (always cache hits)
 - Cache hit ratio 0% (no cache hits)

Attack 3: Locate AES T-Tables

AES T-Table implementation from OpenSSL 1.0.2

- Most addresses in two groups:
 - Cache hit ratio 100% (always cache hits)
 - Cache hit ratio 0% (no cache hits)
- One 4096 byte memory block:
 - Cache hit ratio of 92%
 - Cache hits depend on key value and plaintext value
 - The T-Tables

Attack 4: AES T-Table Template Attack

AES T-Table implementation from OpenSSL 1.0.2

- Known-plaintext attack
- Events: encryption with only one fixed key byte

Attack 4: AES T-Table Template Attack

AES T-Table implementation from OpenSSL 1.0.2

- Known-plaintext attack
- Events: encryption with only one fixed key byte
- Profile each event

Attack 4: AES T-Table Template Attack

AES T-Table implementation from OpenSSL 1.0.2

- Known-plaintext attack
- Events: encryption with only one fixed key byte
- Profile each event
- Exploitation phase:
 - Eliminate key candidates

Attack 4: AES T-Table Template Attack

AES T-Table implementation from OpenSSL 1.0.2

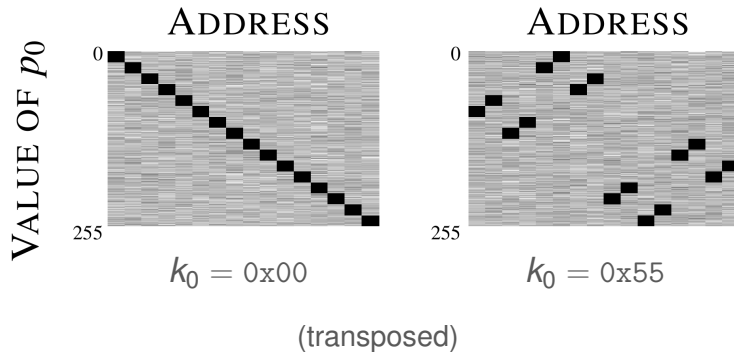
- Known-plaintext attack
- Events: encryption with only one fixed key byte
- Profile each event
- Exploitation phase:
 - Eliminate key candidates
 - Reduction of key space in first-round attack:
 - 64 bits after 16–160 encryptions

Attack 4: AES T-Table Template Attack

AES T-Table implementation from OpenSSL 1.0.2

- Known-plaintext attack
- Events: encryption with only one fixed key byte
- Profile each event
- Exploitation phase:
 - Eliminate key candidates
 - Reduction of key space in first-round attack:
 - 64 bits after 16–160 encryptions
 - State of the art: full key recovery after 30000 encryptions

Attack 4: AES T-Table Template



Conclusion

- Novel technique to find any cache side-channel leakage
 - Attacks
 - Detect vulnerabilities

Conclusion

- Novel technique to find any cache side-channel leakage
 - Attacks
 - Detect vulnerabilities
- Marks a change of perspective:

Conclusion

- Novel technique to find any cache side-channel leakage
 - Attacks
 - Detect vulnerabilities
- Marks a change of perspective:
 - Large scale analysis of (unknown) binaries
 - Large scale automated attacks

Cache Template Attacks:

Automating Attacks on Inclusive Last-Level Caches

Daniel Gruss, Raphael Spreitzer, and Stefan Mangard,
IAIK, Graz University of Technology

August 14, 2015