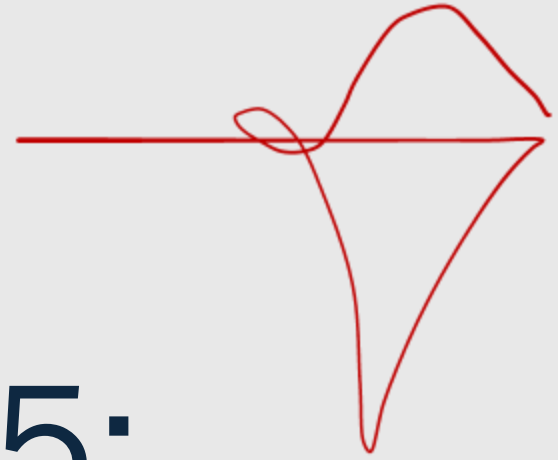


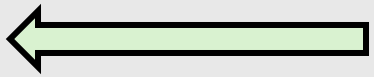
COMP435: *SECURITY CONCEPTS!*

Lecture 27: Rootkits, Trojan Horses, Backdoors, Intro to Privacy

tinyurl.com/comp435-fa25

ROOTKITS, TROJAN HORSES, BACKDOORS

Malware Characteristics

- Attacker's motivation  stealth
- Intended victim
- Structure of malware
- Propagation strategies
- Payload

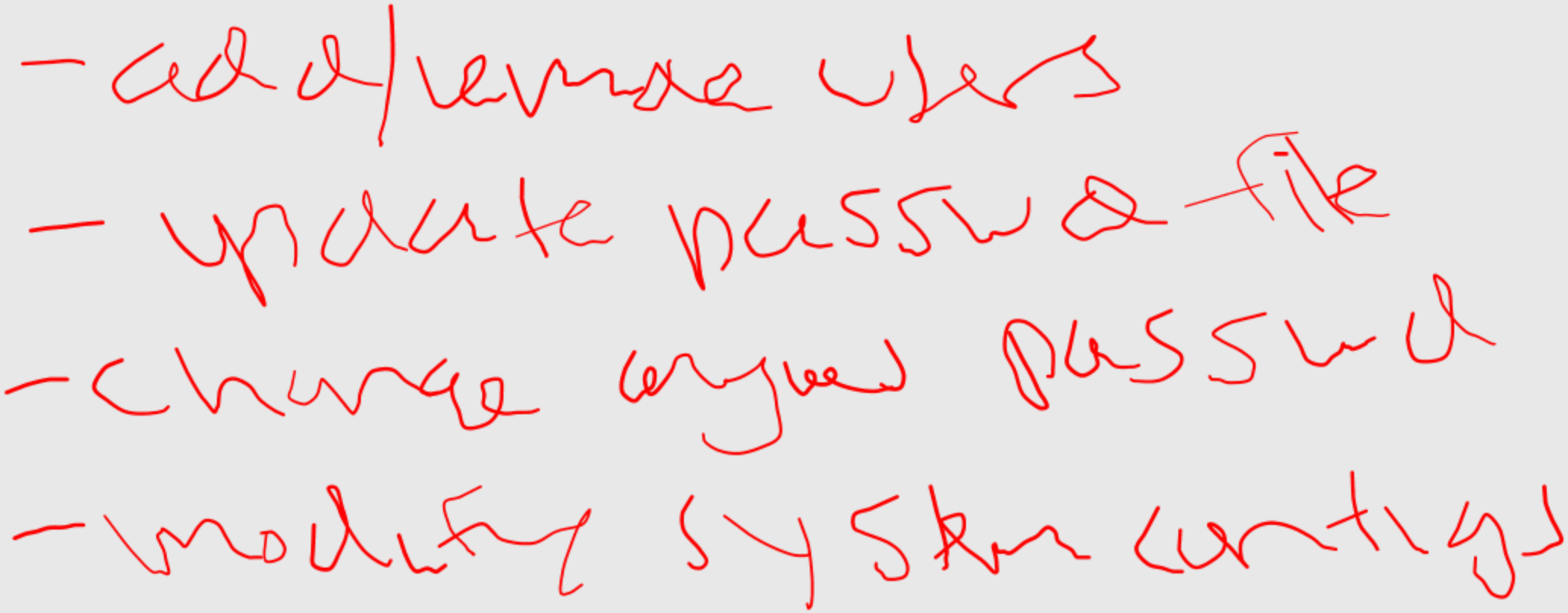
Rootkit

Rootkit

Def’n: a program that obtains privileges of root

Root

Def'n: most privileged subject in a Unix-like operating system



Rootkit

~~Def'n: a program that obtains privileges of root~~

Def'n: a program that compromises and controls kernel software

supervisor mode ≠ root

Rootkit Goals

- Take control early 1
- Remain permanent 2
- Hook into OS subroutines

Hook into OS Subroutines

C:\> dir

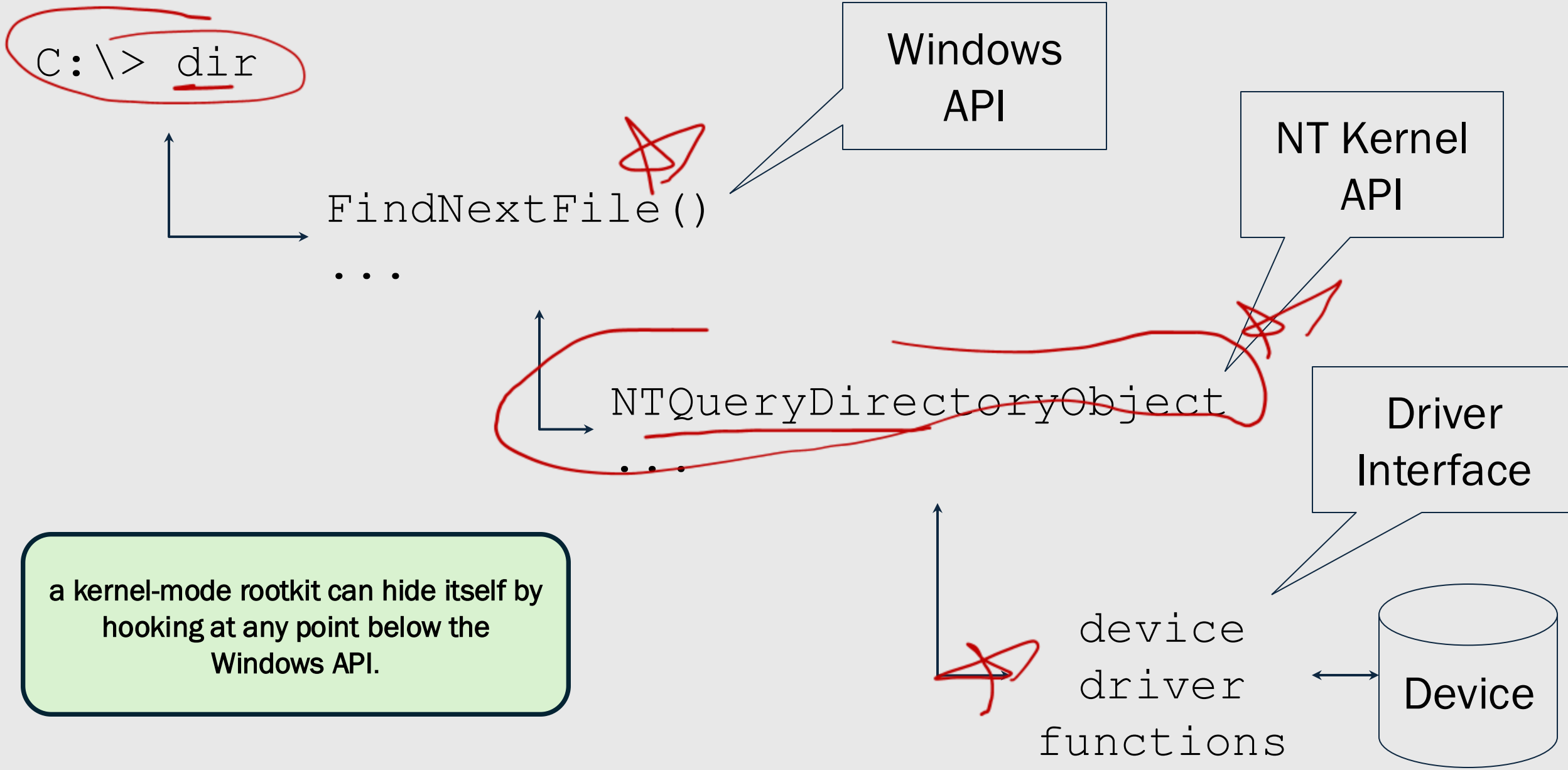
user-mode tools only see what the OS *tells* them.

When you run dir, you're not reading the disk directly, you're calling a high-level API

A kernel-mode rootkit can hide files by intercepting this path

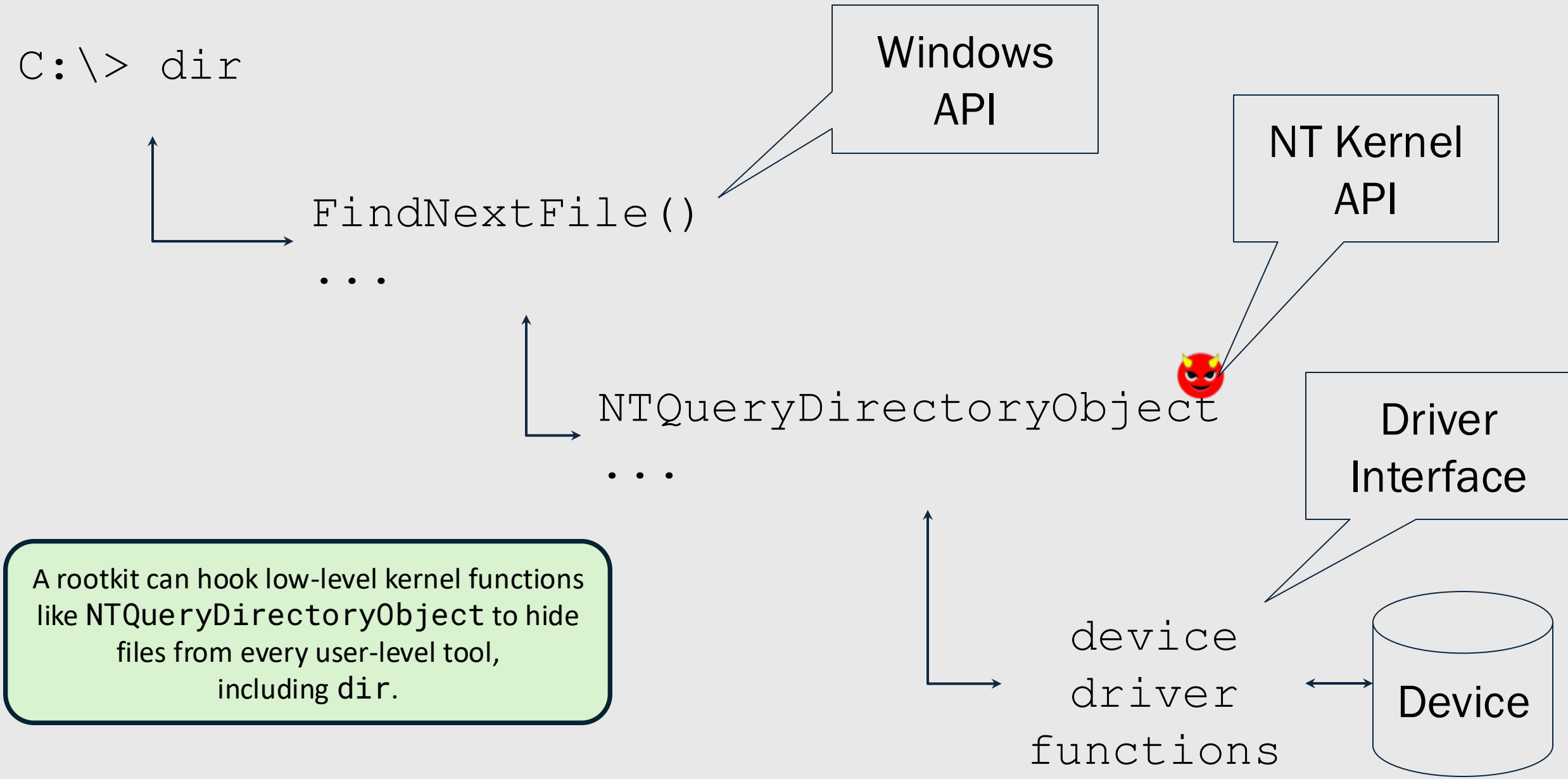
```
C:\Temp> dir
 Volume in drive C is C
 Volume Serial Number is 74F5-B93C

 Directory of C:\Temp

2009-08-25  11:59    <DIR>          .
2009-08-25  11:59    <DIR>          ..
2007-03-01  11:37         2,321,600 AdobeUpdater12345
2009-04-03  10:01            27,988 dd_depcheckdotnet
2009-04-03  10:01               764 dd_dotnetfx3error
2009-04-03  10:01            32,572 dd_dotnetfx3insta
2009-06-09  13:46            35,145 GenProfile.log
2009-08-05  12:11               155 KB969856.log
2009-04-20  08:37               402 MSI29e0b.LOG
2009-04-09  16:34            38,895 offcln11.log
2009-04-03  16:02    <DIR>          OfficePatches
2009-07-14  14:30    <DIR>          OHotfix
2009-08-25  10:52            16,384 Perflib_Perfdata_
2009-04-03  10:01             1,744 uxeventlog.txt
2009-08-25  11:42        50,245,632 WFV2F.tmp
2009-04-20  10:07             1,397 {AC76BA86-7AD7-10
2009-04-20  10:13               617 {AC76BA86-7AD7-10
              13 File(s)     52,723,295 bytes
               4 Dir(s)  83,570,208,768 bytes free
```

"Security in Computing" by Pfleeger, Pfleeger, and Margulies



a kernel-mode rootkit can hide itself by hooking at any point below the Windows API.

"Security in Computing" by Pfleeger, Pfleeger, and Margulies



"Security in Computing" by Pfleeger, Pfleeger, and Margulies

```
C:\> dir
```

Kernel-mode rootkit filters directory listings so its malicious file simply “disappears” from the output of dir

```
EXPLORER.SCF
mal_code.exe
PTDOS.EXE
regedit.exe
TASKMAN.EXE
UNINST32.EXE
```

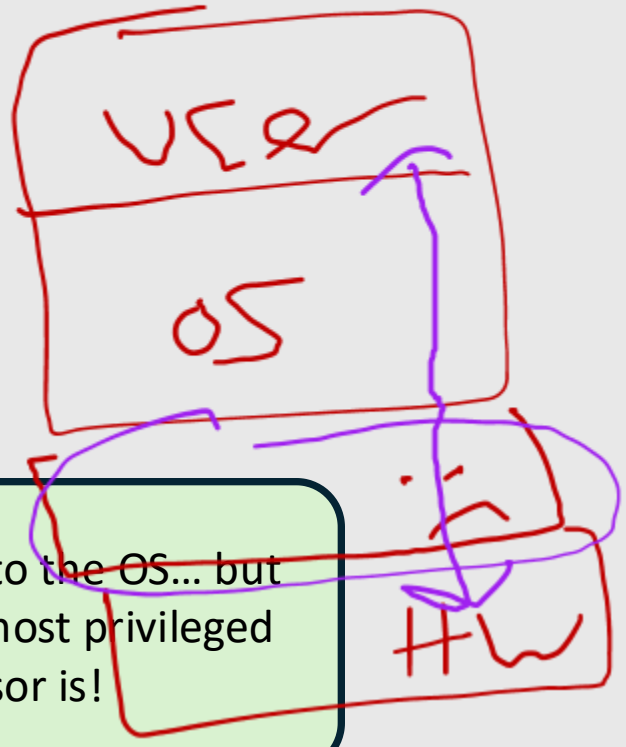
```
EXPLORER.SCF
PTDOS.EXE
regedit.exe
TASKMAN.EXE
UNINST32.EXE
```

“Security in Computing” by Pfleeger, Pfleeger, and Margulies

External Rootkit

Hypervisor: Virtual Machine Monitor (VMM), Virtualizer.. Software that creates and runs virtual machines!

Insert a layer of virtualization under the OS!



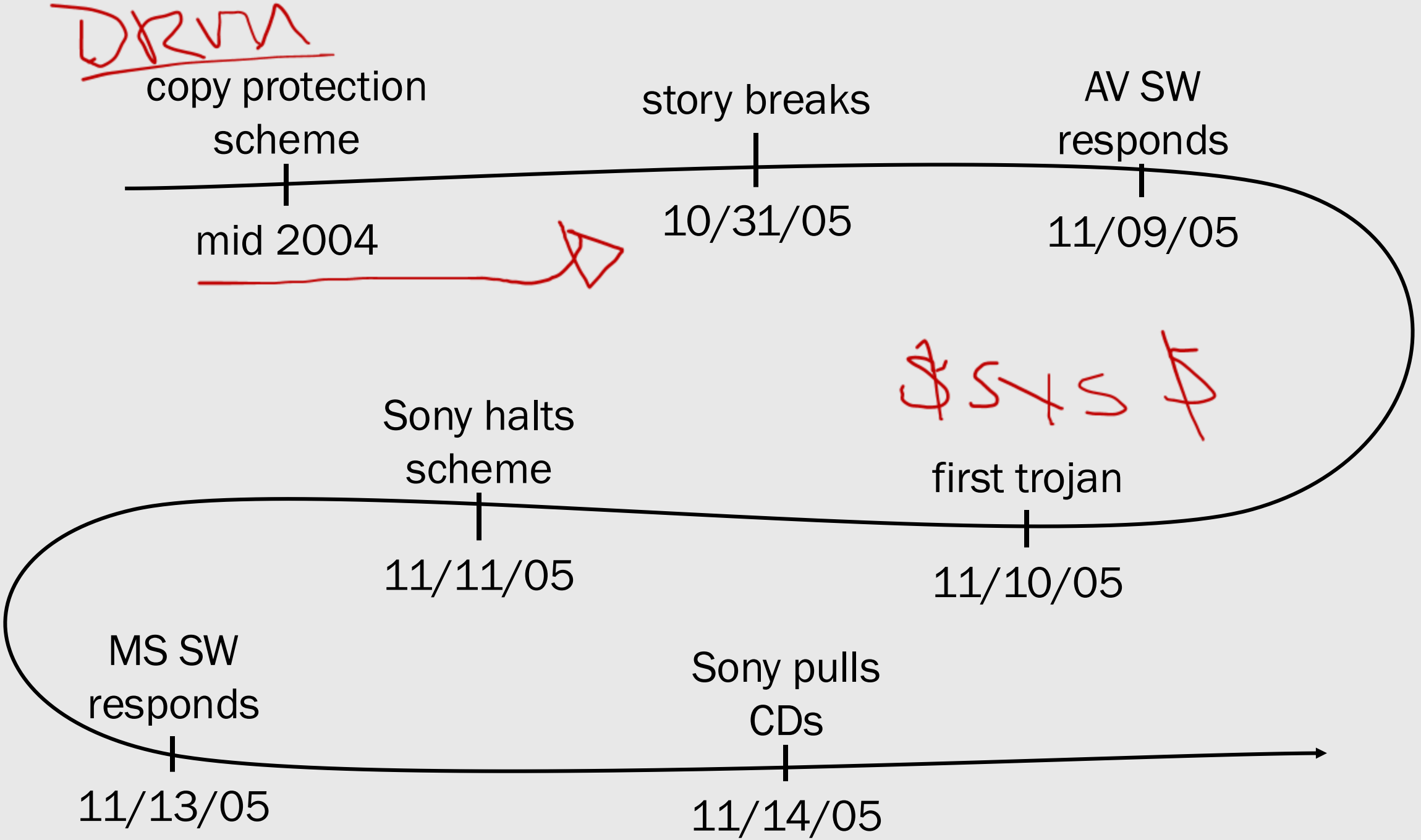
It moves the running OS into a virtual machine/introduces a hidden hypervisor layer… often without the OS realizing it.

Everything will look normal to the OS… but the kernel is no longer the most privileged code… the hypervisor is!

Rootkit Countermeasures

- Secure boot:
  - *ensure the system only boots code that is cryptographically verified to block rootkits from startup*
- Trusted path
  - *Give users a guaranteed, untampered channel to the OS so rootkits cannot spoof prompts or screens*
- Rootkit revealer
  - *Detect inconsistencies between what the OS reports and what's actually on Disk. Ex: fingerprinting*

Sony Rootkit


DRM
copy protection scheme
mid 2004
story breaks
10/31/05
AV SW responds
11/09/05
SYS
first trojan
11/10/05
Sony halts scheme
11/11/05
MS SW responds
11/13/05
Sony pulls CDs
11/14/05

Trojan Horse

Trojan Horse

Def’n: malicious code hidden inside useful or benign code

E.g., screensavers, e-invite greeting cards

Backdoor

Backdoor

Def’n: an entry point into a system that bypasses existing access control

E.g., enable a network service, modify the login process

Advanced Persistent Threat

Advanced Persistent Threat (APT)

- Targeted
- Long-lived
- Well funded
- Stealthy

E.g., Stuxnet, Aurora

PRIVACY

Adapted from:
https://cs.uwaterloo.ca/~m285xu/courses/cs458-w23/assets/modules/intro/slides.pdf

Security vs. Privacy

- In the context of computers… we've been talking about security generally corresponding to:
  - ***C**onfidentiality, **I**ntegrity, **A**vailability*
- Security also has a lot to do w/ "reliability"
  - *Guarantees of C.I.A*

Ex: Keep your personal data confidential

Ex: allow only authorized access or modification to resources

Ex: correct, meaningful results

Privacy

- One definition: “informational self-determination”
  - ***You*** *get to control information about you*
- **Control:**
  - *Who gets to see it*
  - *Who gets to use it*
  - *What they can use it for*
  - *Who they can give it to*

Example: PIPEDA

- PIPEDA (personal information protection and electronic documents act) is Canada's private-sector privacy legislation
- Lists 10 fair information principles companies need to abide by:

1. *Identify the purpose of data collection*
2. *Obtain consent*
3. *Limit collection*
4. *Limit use, disclosure and retention*
5. *Use appropriate safeguards*
6. *Give individuals access*
7. *Be accurate*
8. *Be open*
9. *Be accountable*
10. *Provide recourse*

Read more: https://www.priv.gc.ca/leg_c/p_principle_e.asp

Consumer Privacy Protection Act

- Forthcoming legislation to regulate private sector use of personal information
- Modernizing protection: meaningful consent, right to erasure, etc
- Stronger provisions for enforcement
- Private right of action

Security vs. Privacy

- Some place security & privacy as if they're opposing forces…
- Are they? Do we have to give up one to get the other?

DATA SECURITY & PRIVACY

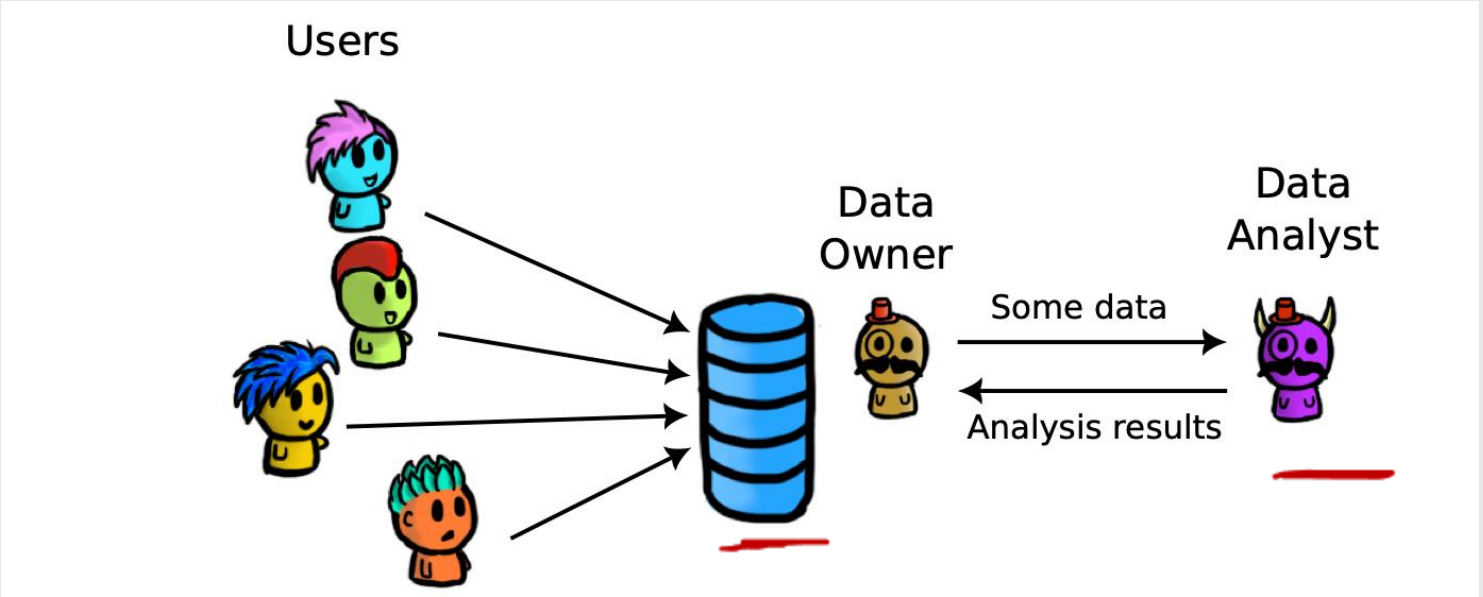
Adapted from:
https://cs.uwaterloo.ca/~m285xu/courses/cs458-w23/assets/modules/intro/slides.pdf

Recall: Database Security Requirements

- Access Control
  - *Who can read? Who can write?*
- Authentication
  - *How do we know if a DB client is not pretending to be someone else*
- Confidentiality
  - *What if the DB server is compromised? What about network tapping/eavesdropping?*
- Integrity
  - *How do we guarantee the data is intact and correct?*
- Availability
  - *Redundancy? Fault-tolerance?*
- Auditability
  - *Proving/providing evidence of how we ended up in a specific state*

System Model: Privacy + Utility Tradeoff..

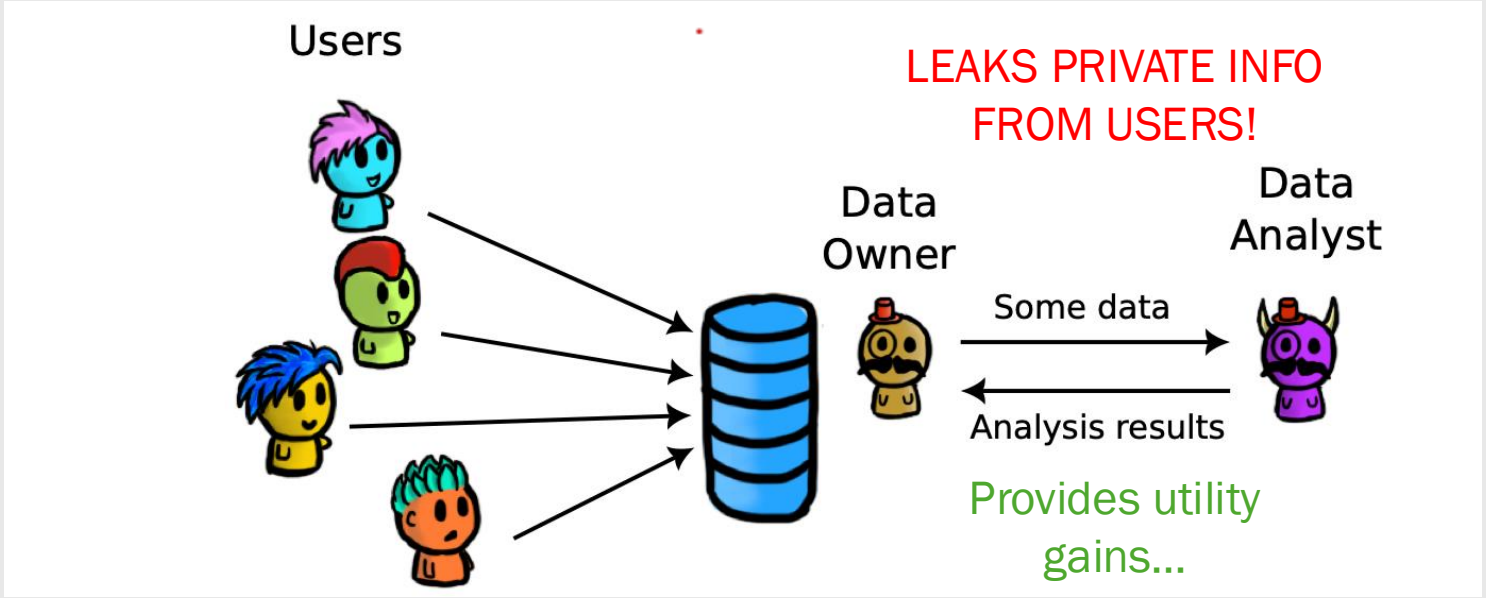
Users provide their data to a data pool.
Admin shares ***a slice of data*** in the pool with a data analyst



https://cs.uwaterloo.ca/~m285xu/courses/cs458-w23/assets/modules/data-sec/part-2-inference-and-syntactic-notions.pdf

System Model: Privacy + Utility Tradeoff..

Users provide their data to a data pool.
Admin shares ***a slice of data*** in the pool with a data analyst

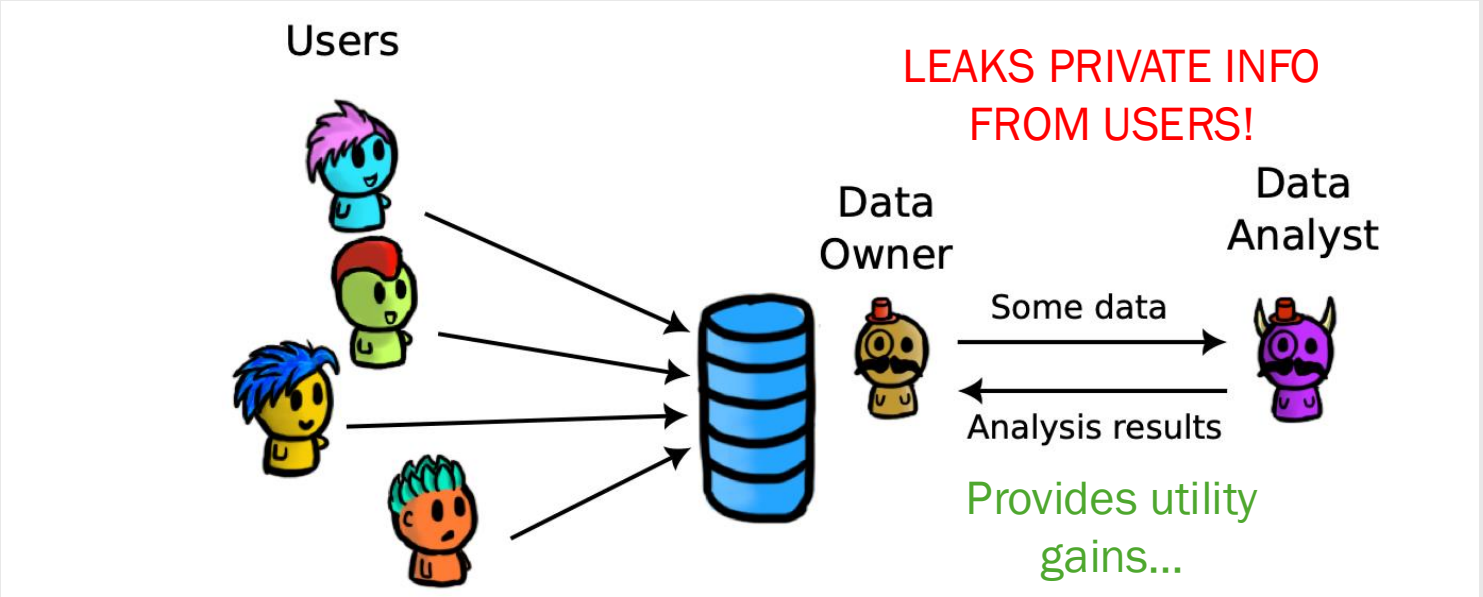


Utility: benefits for both users or the data owner/service provider

Privacy: important for users, it's their data + their right to privacy!

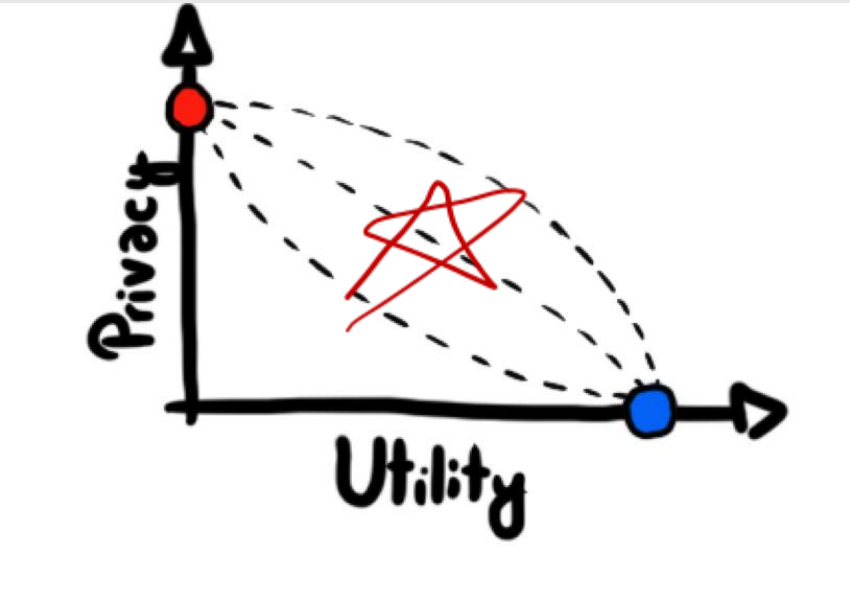
System Model: Privacy + Utility Tradeoff..

Users provide their data to a data pool.
Admin shares ***a slice of data*** in the pool with a data analyst



Q: Any concrete examples that fit this model?

System Model: Privacy + Utility Tradeoff..



Q: easy approaches that fit the red/blue points?

Finding good mechanisms in the middle is hard!!

Some metrics

- There is no perfect metric for privacy and utility that works for every scenario
- **Syntactic notions of privacy** (properties that the published data must follow)
  - k-anonymity
  - ℓ-diversity
  - t-closeness

Some metrics

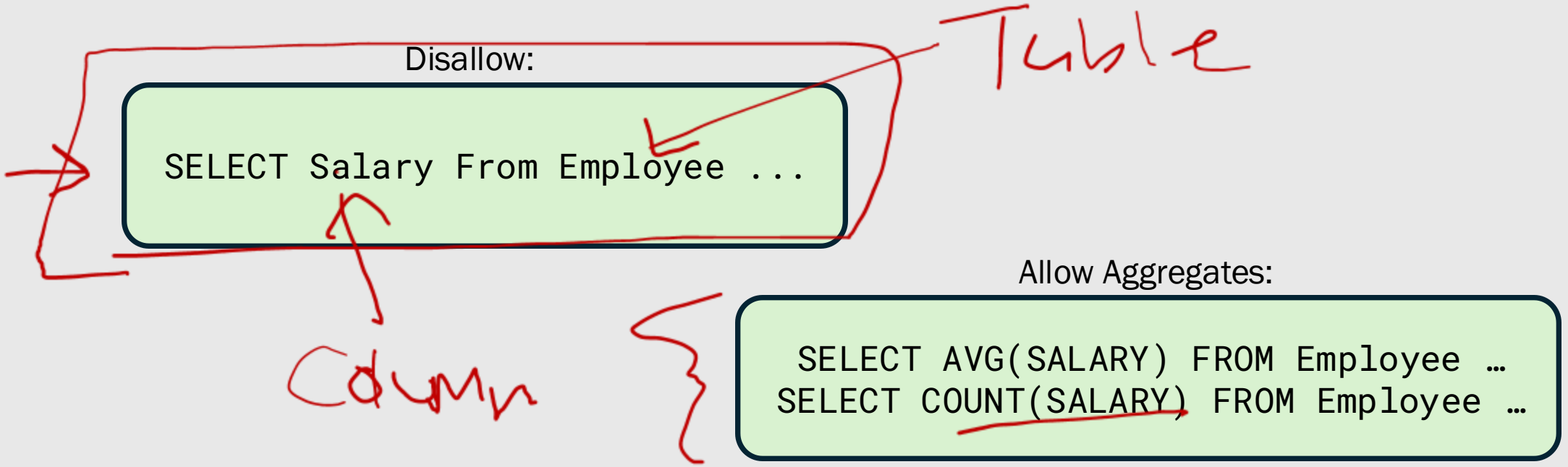
- There is no perfect metric for privacy and utility that works for every scenario
- **Syntactic notions of privacy** (properties that the published data must follow)
  - k-anonymity
  - ℓ-diversity
  - t-closeness
- **Semantic notions of privacy** (properties that the data release mechanism must follow / this is independent of the data that is actually published)
  - **Differential privacy!**

An example: SQL queries

- Scenario: large db with some sensitive attributes
- Utility: we want to allow certain queries
  - Ex: get avg salary of everyone in the company
- Privacy: We also want to protect the privacy of people whose data is in the db
  - Ex: compute avg without revealing each individual's salary

An example: SQL queries

- Scenario: large db with some sensitive attributes
- Utility: we want to allow certain queries
  - Ex: get avg salary of everyone in the company
- Privacy: We also want to protect the privacy of people whose data is in the db
  - Ex: compute avg without revealing each individual's salary



Data inference problem

- Data analysts could **infer** sensitive data, through the output of allowed aggregate queries
  - Inference doesn't have to be a full or accurate recovery either!
  - Ex: determining the range of a specific employee's salary would be considered a leak

- **Goal:** minimize (unintentional) leaks of sensitive data to the data analysts through the allowed queries!

Disallow:

```
SELECT Salary From Employee ...
```

Allow Aggregates:

```
SELECT AVG(SALARY) FROM Employee …
SELECT COUNT(SALARY) FROM Employee …
```

Inference Attack: Single Query

- One single query directly outputs the sensitive data

```
SELECT SUM(Salary) FROM Employee
  WHERE Name = "Alice"
  AND (Gender = "M" OR Gender = "F" OR Gender = "X");
```

Inference Attack: Single Query

- One single query directly outputs the sensitive data

```
SELECT SUM(Salary) FROM Employee
    WHERE Name = "Alice"
    AND (Gender = "M" OR Gender = "F" OR Gender = ”X");
```

Countermeasure: If the SELECT clause output includes less than k results, then drop the query. k is usually application specific.

Inference Attack: Multiple Queries

- How can you infer Alice's Salary in this case?

Name (PK)	Age	Zip	Salary
Alice	32	N2L 0G7	55 000 CAD
Bob	34	N2L 3E4	65 000 CAD
Carol	26	N2L 0E1	35 000 CAD
Dave	24	N2L 2W4	40 000 CAD
...			

Table: Employee (example only)

Inference Attack: Multiple Queries

- How can you infer Alice's Salary in this case?

Name (PK)	Age	Zip	Salary
Alice	32	N2L 0G7	55 000 CAD
Bob	34	N2L 3E4	65 000 CAD
Carol	26	N2L 0E1	35 000 CAD
Dave	24	N2L 2W4	40 000 CAD
...			

Table: Employee (example only)

We can use set theory!

Indirect attack

Q_1: `SELECT SUM(Salary) FROM Employee;` (outputs s)

Q_2: `SELECT SUM(Salary) FROM Employee WHERE Name != "Alice";` (outputs r)

$s - r$ reveals the secret salary.

Inference Attack: Multiple Queries

- How can you infer Alice's Salary in this case?

Name (PK)	Age	Zip	Salary
Alice	32	N2L 0G7	55 000 CAD
Bob	34	N2L 3E4	65 000 CAD
Carol	26	N2L 0E1	35 000 CAD
Dave	24	N2L 2W4	40 000 CAD
...			

Table: Employee (example only)

Countermeasure: Suppose the database has a total of N records. If the SELECT clause output includes less than k results, or more than N−k results (but less than N results), then drop the query. NOTE: a query that includes N records (i.e., all records) is OK.