

# SymLink Races and how to deal with them

## Storage Security Summit 2022

Volker Lendecke

Samba Team / SerNet

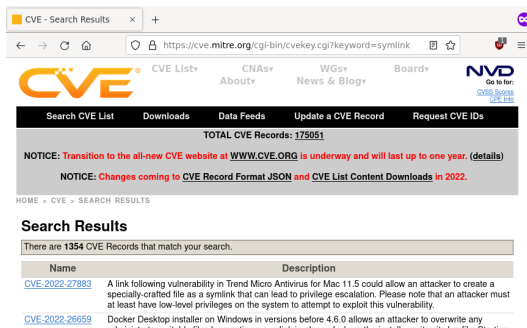
2022-05-11

# Who am I?

- ▶ Co-Founder of SerNet in Göttingen, Germany
- ▶ First Samba patches in 1994
- ▶ Early Samba Team member
- ▶ Samba infrastructure (tdb, tevent, etc)
- ▶ File server
- ▶ Clustered Samba
- ▶ Winbind
- ▶ AD controller is my colleague Stefan Metzmacher's domain
  - ▶ Stefan implemented AD multi-master replication in Samba

# Symlink race – Why care?

- Search for “symlink” in Common Vulnerabilities and Exposures:



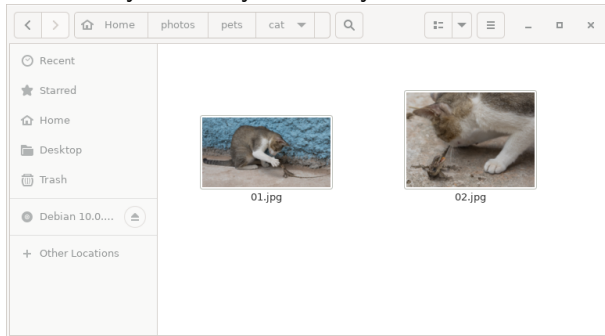
The screenshot shows a web browser window with the URL <https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=symlink>. The page is titled "CVE - Search Results" and displays the CVE Mitre website header. The main content area shows "TOTAL CVE Records: 175051" and two notices: "Transition to the all-new CVE website at [WWW.CVE.ORG](http://WWW.CVE.ORG) is underway and will last up to one year." and "Changes coming to CVE Record Format JSON and CVE List Content Downloads in 2022." Below the notices, the "Search Results" section indicates "There are 1354 CVE Records that match your search." and lists two results:

Name	Description
<a href="#">CVE-2022-27883</a>	A link following vulnerability in Trend Micro Antivirus for Mac 11.5 could allow an attacker to create a specially-crafted file as a symlink that can lead to privilege escalation. Please note that an attacker must at least have low-level privileges on the system to attempt to exploit this vulnerability.
<a href="#">CVE-2022-26659</a>	Docker Desktop installer on Windows in versions before 4.6.0 allows an attacker to overwrite any administrator writable files by creating a symlink in place of where the installer writes its log file. Starting

- Ok, there's something going on – but what??

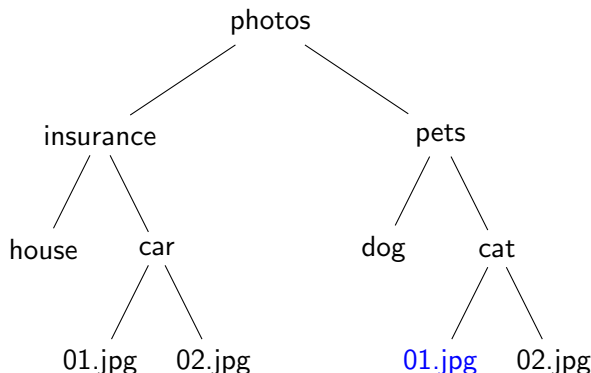
# Publish cat photos to my blog

## ► Browse my directory hierarchy



- Copy file path `/photos/pets/cat/01.jpg` into Browser
- Press upload

# My directory hierarchy



- Cute cat under `/photos/pets/cat/01.jpg`

Cute cat

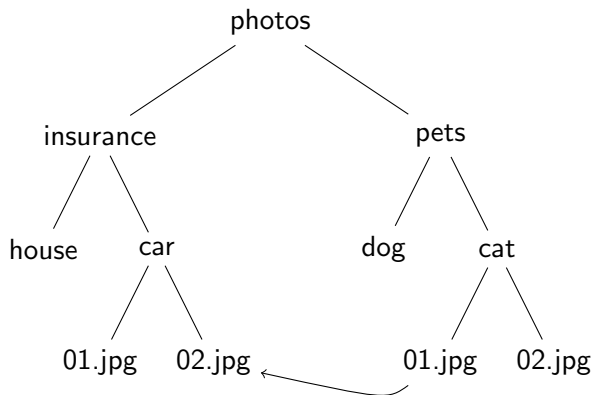
▶ /photos/pets/cat/01.jpg



# Symlink Race

- ▶ Publish cat photo
  - ▶ Browse my directory hierarchy
  - ▶ Copy file path /photos/pets/cat/01.jpg into Browser
    - ▶ A few seconds for an **ATTACK**
  - ▶ Press upload
- ▶ The attacker replaces “01.jpg” with a symbolic link to the latest communication with my insurance
- ▶ In -sf /photos/insurance/car/02.jpg /photos/pets/cat/01.jpg

# My new directory hierarchy





# What gets published?

- ▶ `/photos/pets/cat/01.jpg` gets uploaded, but... oops



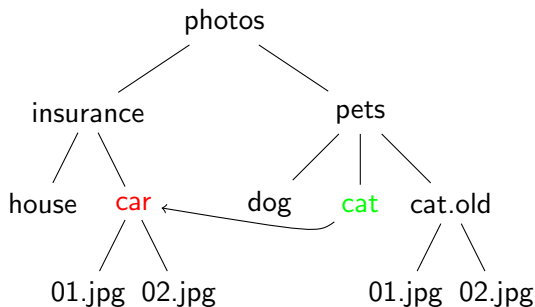
- ▶ This could have been your `Passwords.DOCX`

# O\_NOFOLLOW

- ▶ Posix/Linux can protect against this kind of attack
- ▶ Opening a file in Posix via C function `open()`
- ▶ Adding the flag `O_NOFOLLOW` to `open()` ensures “01.jpg” is not a symlink:
  - ▶ If the trailing component (i.e., `basename`) of `pathname` is a symbolic link, then the `open` fails, with the error `ELOOP`
- ▶ When using `O_NOFOLLOW`, the upload function will get an error
- ▶ However:
  - ▶ Symbolic links in earlier components of the `pathname` will still be followed.

# My latest directory hierarchy

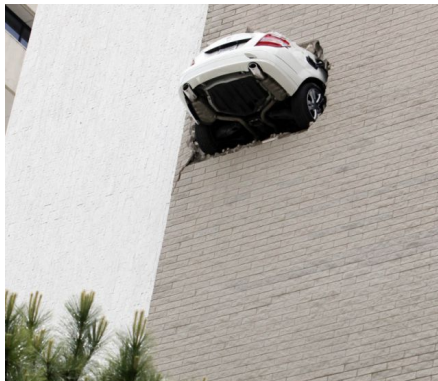
## ► Another **attack**



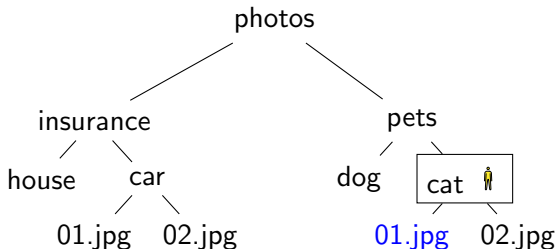
- `/photos/pets/cat` → `/photos/insurance/car`
- `/photos/pets/cat/01.jpg` → `/photos/insurance/car/01.jpg`

# What gets published?

- ▶ `/photos/pets/cat/01.jpg` gets uploaded, but... ouch!

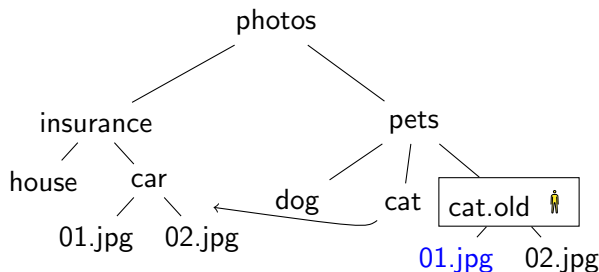


## Change into /photos/pets/cat



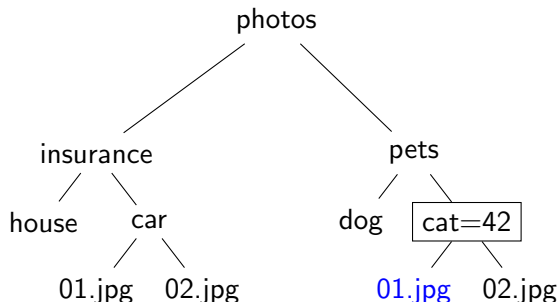
- ▶ Four expensive steps
  - ▶ `chdir /photos/pets/cat`
  - ▶ check I am really in `/photos/pets/cat`
  - ▶ `open("01.jpg", O_NOFOLLOW)`
  - ▶ `chdir /`

# Attack mitigated



- ▶ Attacker moves me to cat.old
- ▶ `open("01.jpg", O_NOFOLLOW)` references the correct file
- ▶ Opening files becomes expensive due to the four steps

# Openat-Call



- ▶ Four expensive steps done just once
- ▶ Hold and cache reference to /photos/pets/cat (=42)
- ▶ Cheap `openat(42, "01.jpg", O_NOFOLLOW)`

# Symlink races from 30,000ft

- ▶ Paths in Posix are prone to input validation problems
- ▶ Symlink races are a Time-Of-Check Time-Of-Use (TOCTOU) problem
- ▶ Sanitizing paths is possible, but tedious  
⇒ Nobody does it
- ▶ Current status in Samba with 4.15:
  - ▶ Many places in Samba choose the chdir/check/open/chdir way
  - ▶ Work is ongoing to pass directory handles (the “42”) everywhere
- ▶ Work ongoing in Linux to make sanitizing easier:
  - ▶ `openat2(RESOLVE_NO_SYMLINKS)`
  - ▶ Mount option disallowing symlinks?



# Thanks for your attention

vl@samba.org / vl@sernet.de  
<https://www.sernet.de/>  
<https://www.samba.org/>