Code signing flaws in macOS

> whoami

Thomas Reed
Director of
Mac & Mobile
@ Malwarebytes

@thomasareed

Old-school malware

> rc.common persistence

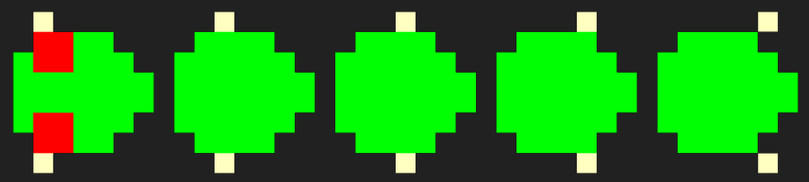
> Add malicious code to the end of /etc/rc.common

> MacDownloader unused code:

```
> if cat /etc/rc.common | grep %@; then
  sleep 1; else echo 'sleep %d && %@ &' >>
  /etc/rc.common; fi
```

> Does not work in Lion (OS X 10.7) or higher

Old-school malware



> cron

> Found in recent VSearch (aka Pirrit) variant

> $ sudo crontab -l
  50 * * * * /Library/stateliness.hu/
  stateliness.hu cr

Old-school malware

> Viruses!?

> "A virus operates by inserting or attaching itself to a legitimate program or document [...] in order to execute its code."[1]

> None currently active on Macs

> Cases where malware was added to an existing app were done manually, not automatically

1 - https://us.norton.com/internetsecurity-malware-what-is-a-computer-virus.html

Mac viruses?

> Code signing theory

> Cryptographically sign an app with private key

> If app is modified, code signature becomes invalid

> App with invalid signature should not be allowed to run

Mac viruses?

> Apple tools that give information about code signatures:

  > codesign

  > spctl

> Third-party apps:

  > What's Your Sign? (Objective-See.com)

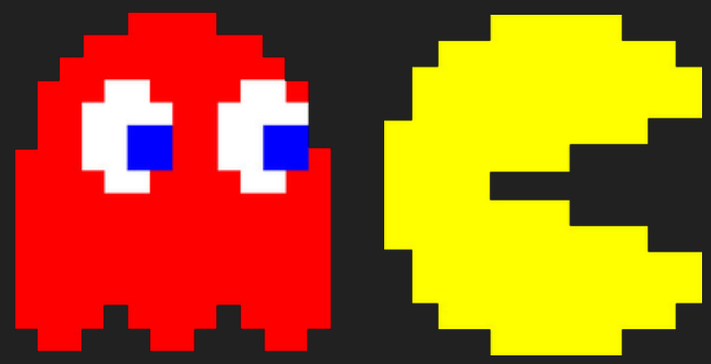
  > RB App Checker Lite (Mac App Store)

Mac viruses?

> Are viruses impossible on modern macOS, due to code signing?

> Unfortunately, no.

> Why not? Let's look at how code signing works.

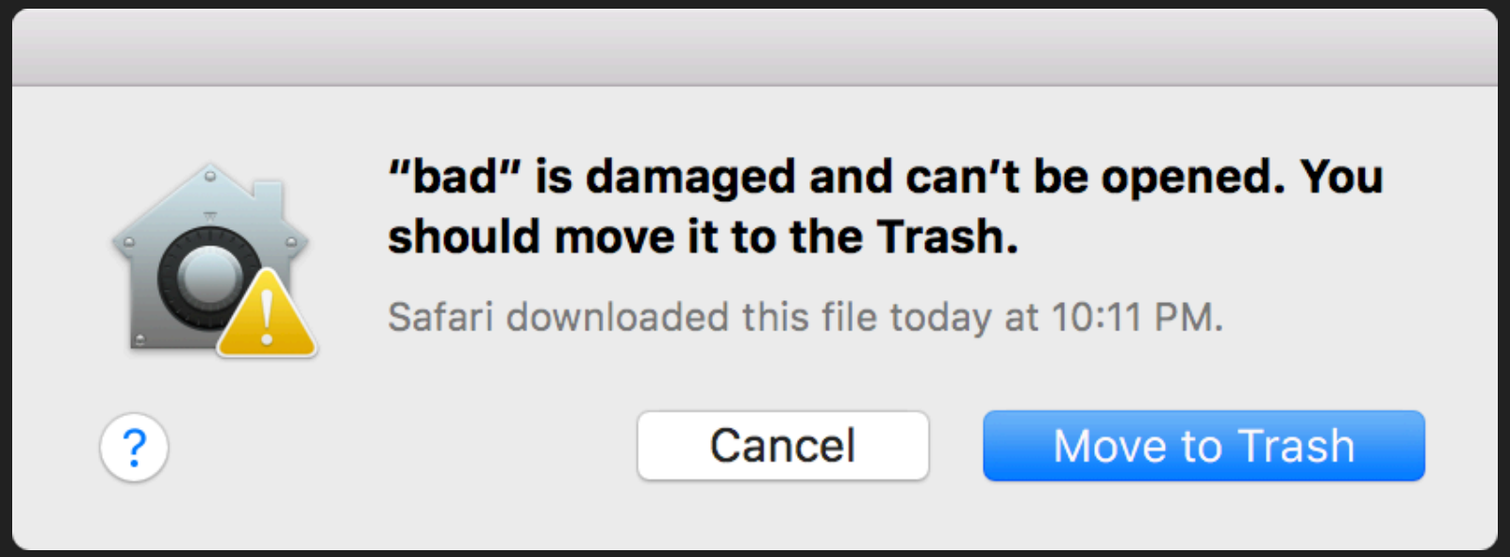


Code signing on Mac

> Most apps code signed today

> Unsigned apps not allowed by default

> macOS verifies code signature before running downloaded apps

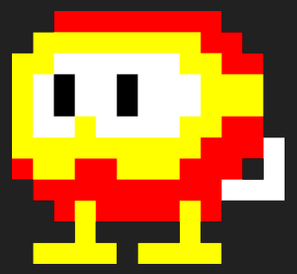


Code signing on Mac

> So, code signed apps are safe, right?

WRONG!

Code signing on Mac



> Apps are "quarantined" when downloaded

> Gatekeeper only checks code signature for quarantined apps

> After opening, quarantine flag is removed

> Code signature is never checked again!

Time for mischief!

> Malware almost always wants persistence

> Malware almost always wants to be hidden

> Achieve both by infecting apps that are no longer quarantined!

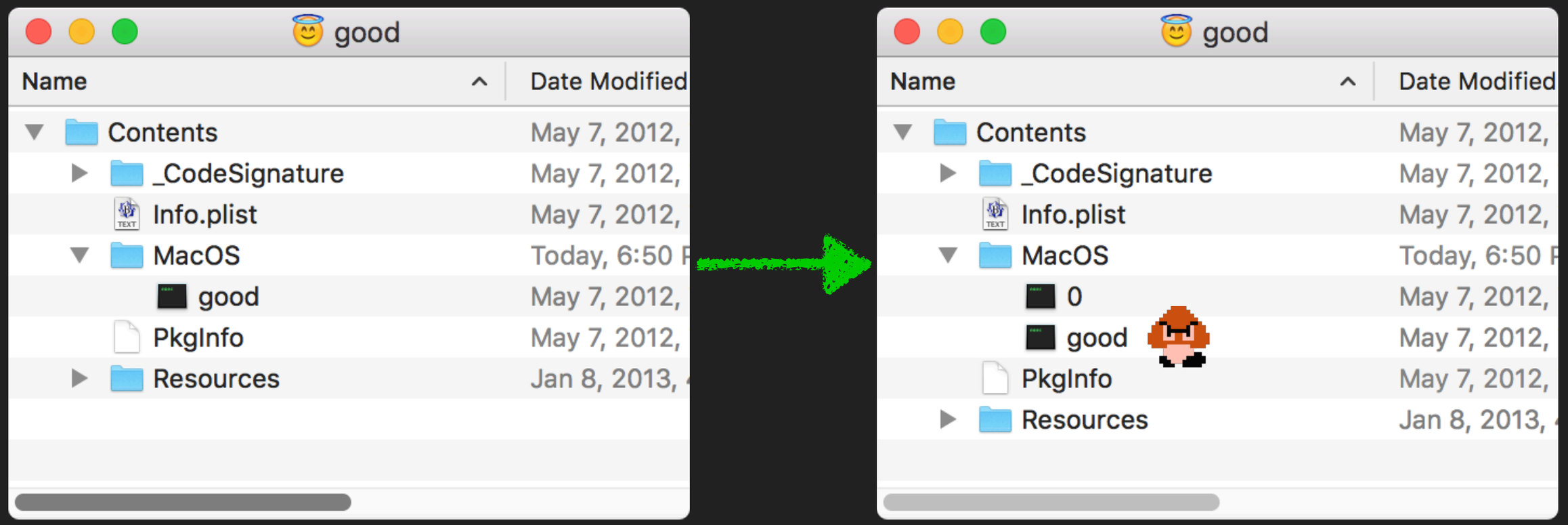
> Malicious code will run every time an infected app is opened

Infecting an app

> Rename "good" to something else - like "0"

> Add malicious executable named "good"

> "good" executable loads "0" to make the app seem normal



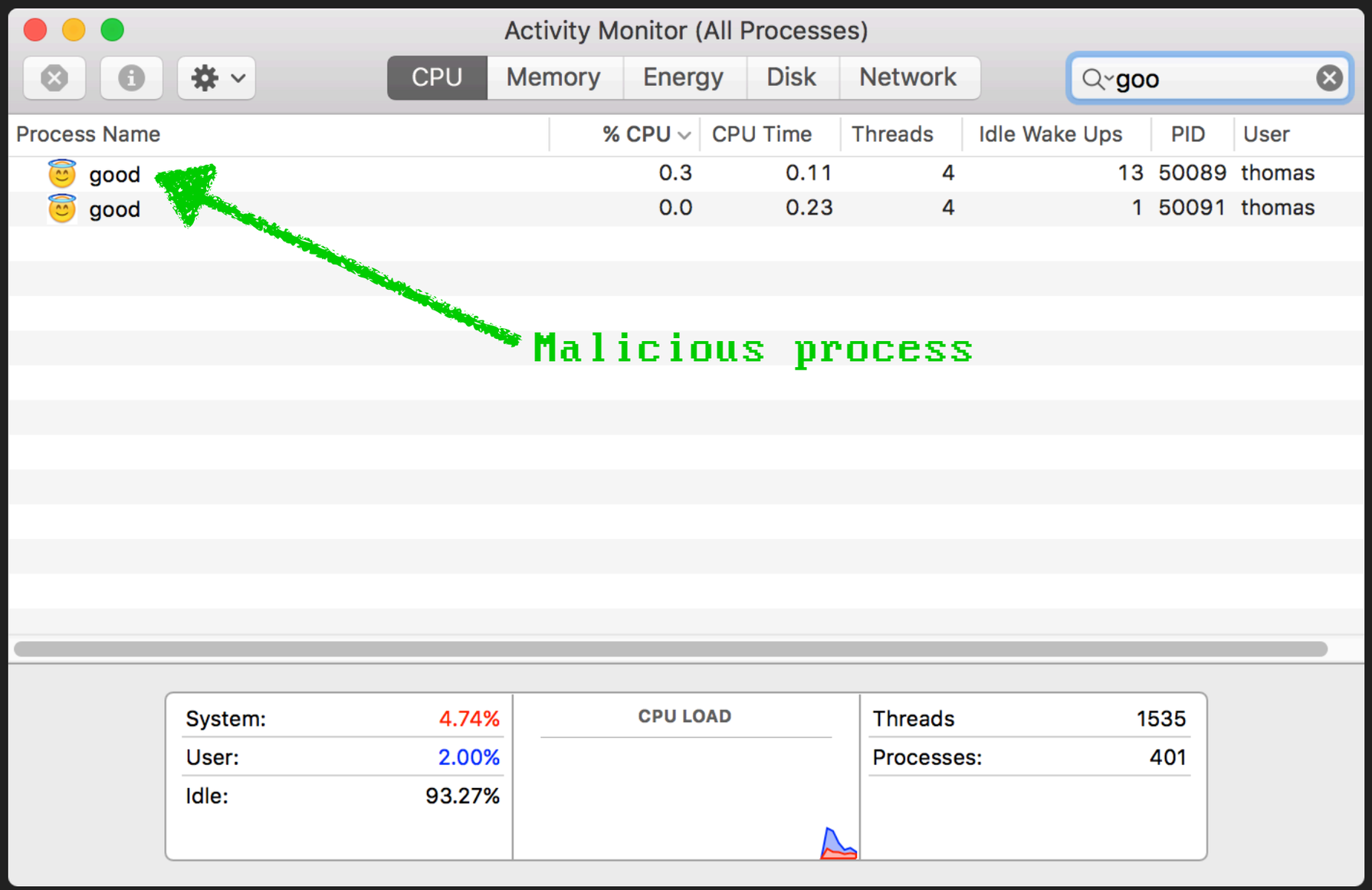
Infecting an app

User double-clicks "good" app

System opens "good" executable file (which is actually malicious)

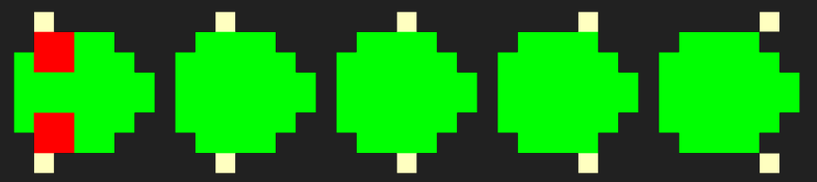
"good" executable opens original, renamed executable, to avoid suspicion

Infecting an app



You have dysentery.

How hard is this?



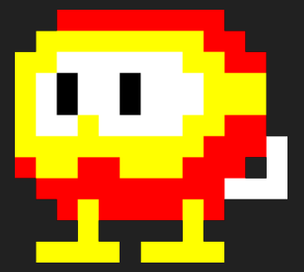
> Not very!

> 22 lines of Swift code - malicious executable

> 18 lines of AppleScript - dropper part 1

> 16 lines of shell script - dropper part 2

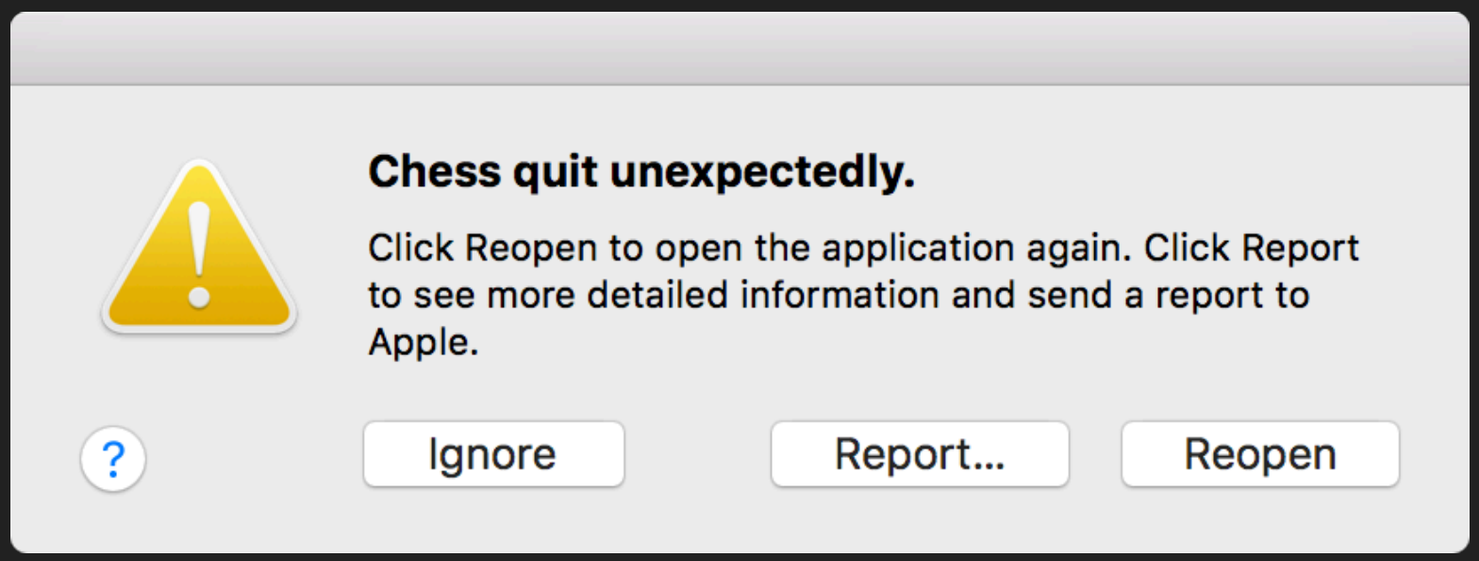
Exceptions



> Apple's apps can't be modified

> If you try it, they crash

> Malicious code still runs!

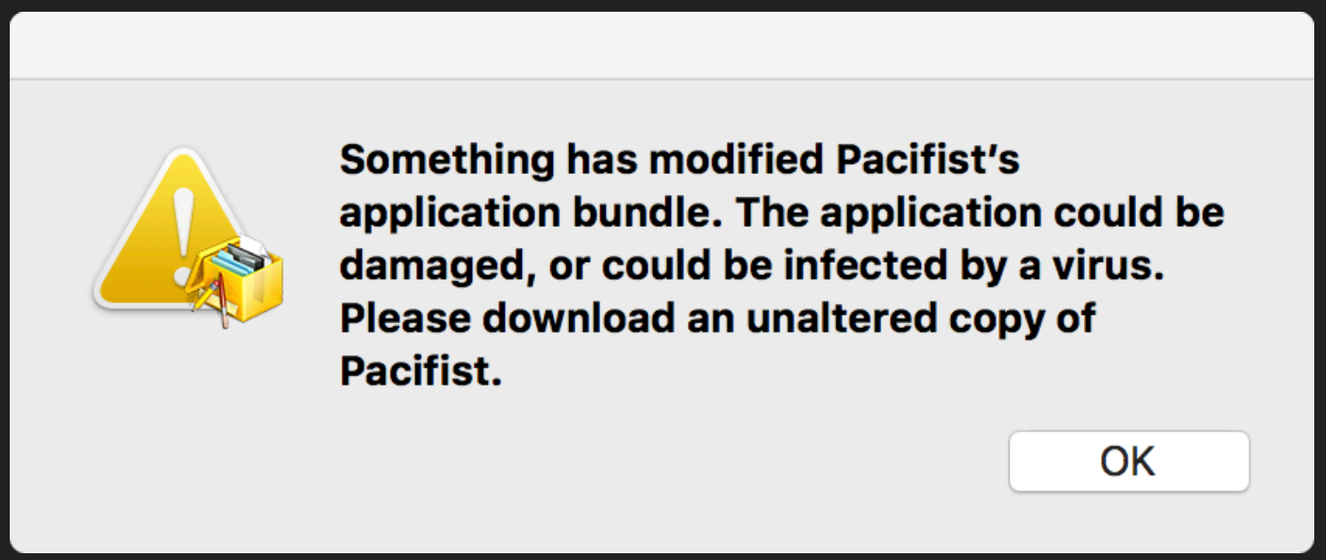


Exceptions

> Some third-party apps have self-protection

> If you change them, they'll let the user know

> Malicious code still runs!



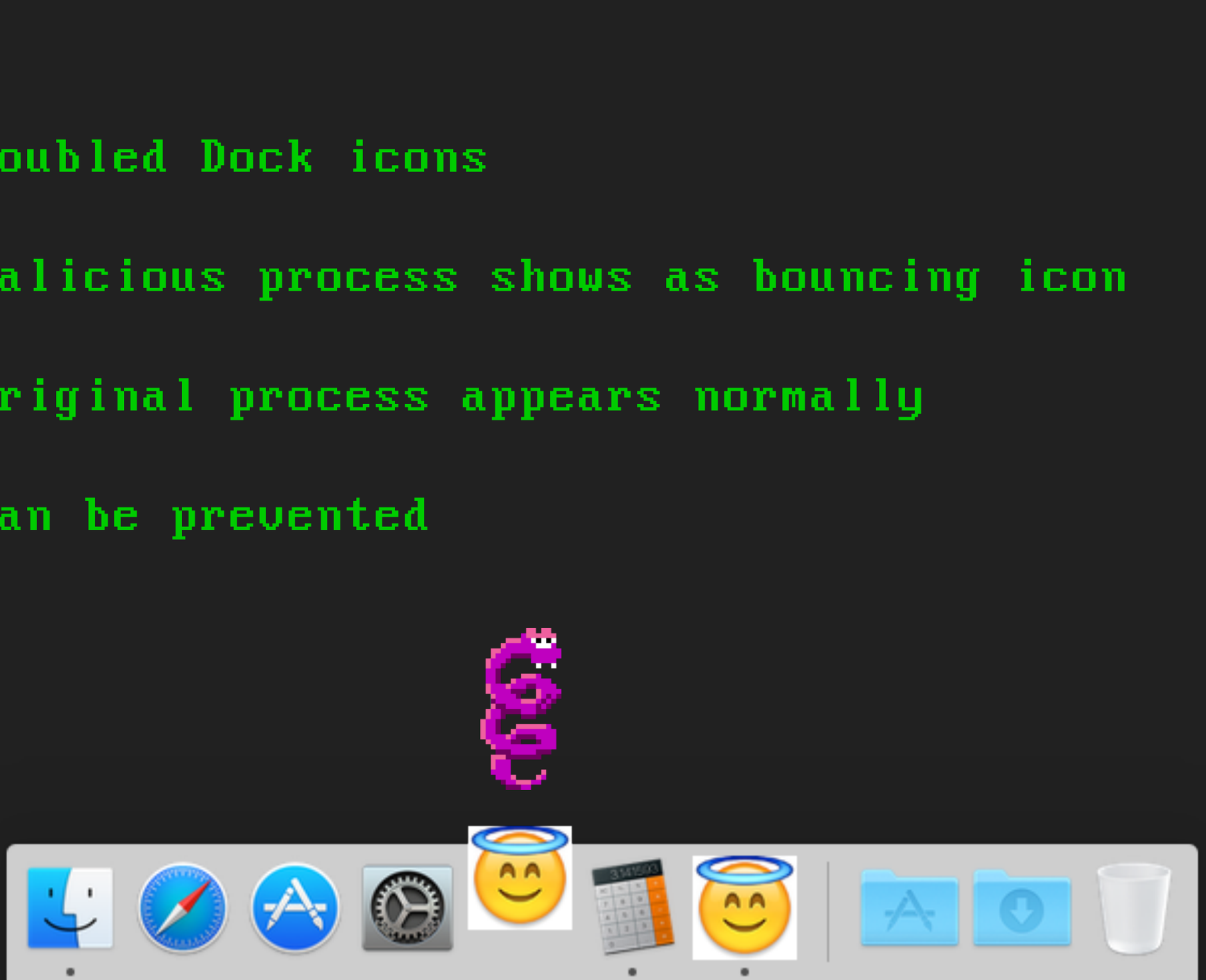
Potential giveaways

> Doubled Dock icons

> Malicious process shows as bouncing icon

> Original process appears normally

> Can be prevented

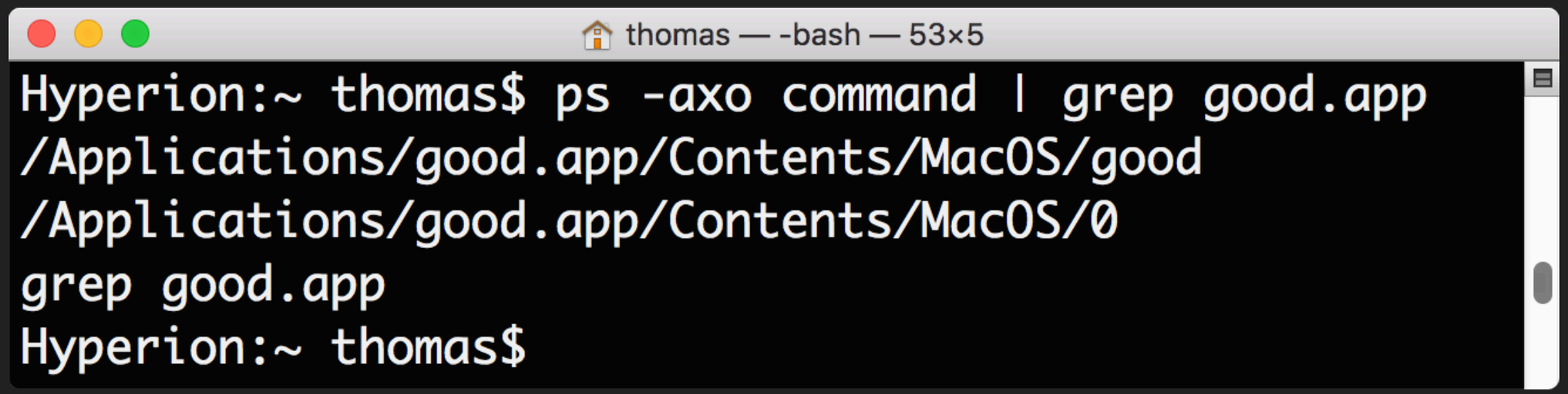


Potential giveaways

> Two processes in Activity Monitor

> Two processes in ps output

> Could make this less suspicious fairly easily



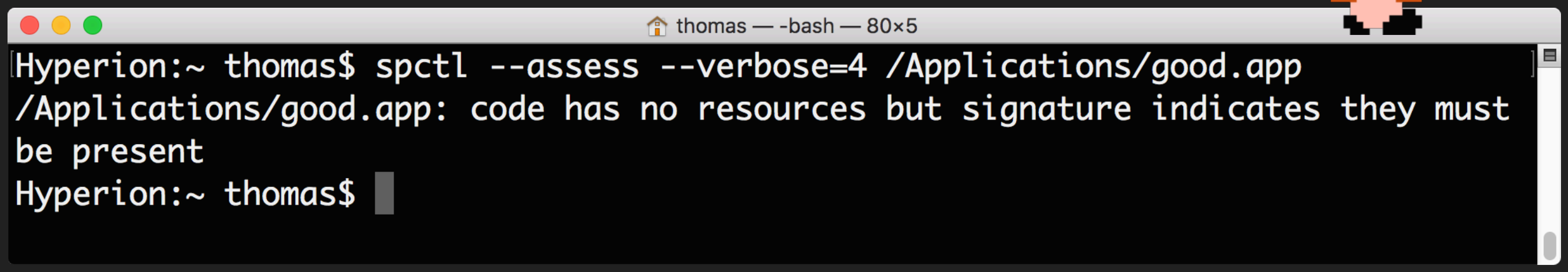
Demo time...

How to detect

> Use spctl to verify signature

Good signature:

Bad signature:



How to detect

> Use osquery to check signature

thomas — -bash — 80×10

```
Hyperion:~ thomas$ osqueryi --line 'select * from signature where path="/Applica
tions/good.app"'
           path = /Applications/good.app
         signed = 1
     identifier = com.thesafemac.good
         cdhash = 7bdebb6406c5148f5b055971af11864da4633d40
team_identifier = DKYMKWTFCU
      authority = Developer ID Application: Thomas Reed (DKYMKWTFCU)
Hyperion:~ thomas$
```

How to detect

> Use osquery to check signature

thomas — -bash — 80×10

```
Hyperion:~ thomas$ osqueryi --line 'select * from signature where path="/Applica
tions/good.app"'
          path = /Applications/good.app
        signed = 0
    identifier = MalTest-55554944f1486d7a59f535fabef31b690dbbc92b
        cdhash = 93c51b7dc46ca82eba78e00b93685ccb914757b6
team_identifier =
     authority =
Hyperion:~ thomas$
```

Problem...

> What if the dropper re-signs the app with a different certificate?

thomas — -bash — 80×9

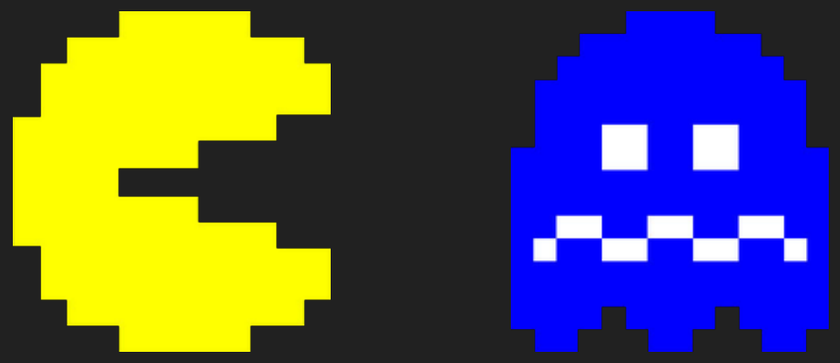
```
Hyperion:~ thomas$ osqueryi --line 'select * from signature where path="/Applica
tions/good.app"'
          path = /Applications/good.app
        signed = 1
    identifier = com.goomba.good
        cdhash = 93c51b7dc46ca82eba78e00b93685ccb914757b6
team_identifier = ABCDEFGHIJ
     authority = Developer ID Application: Joe Smith (ABCDEFGHIJ)
Hyperion:~ thomas$
```

Naughty or nice?

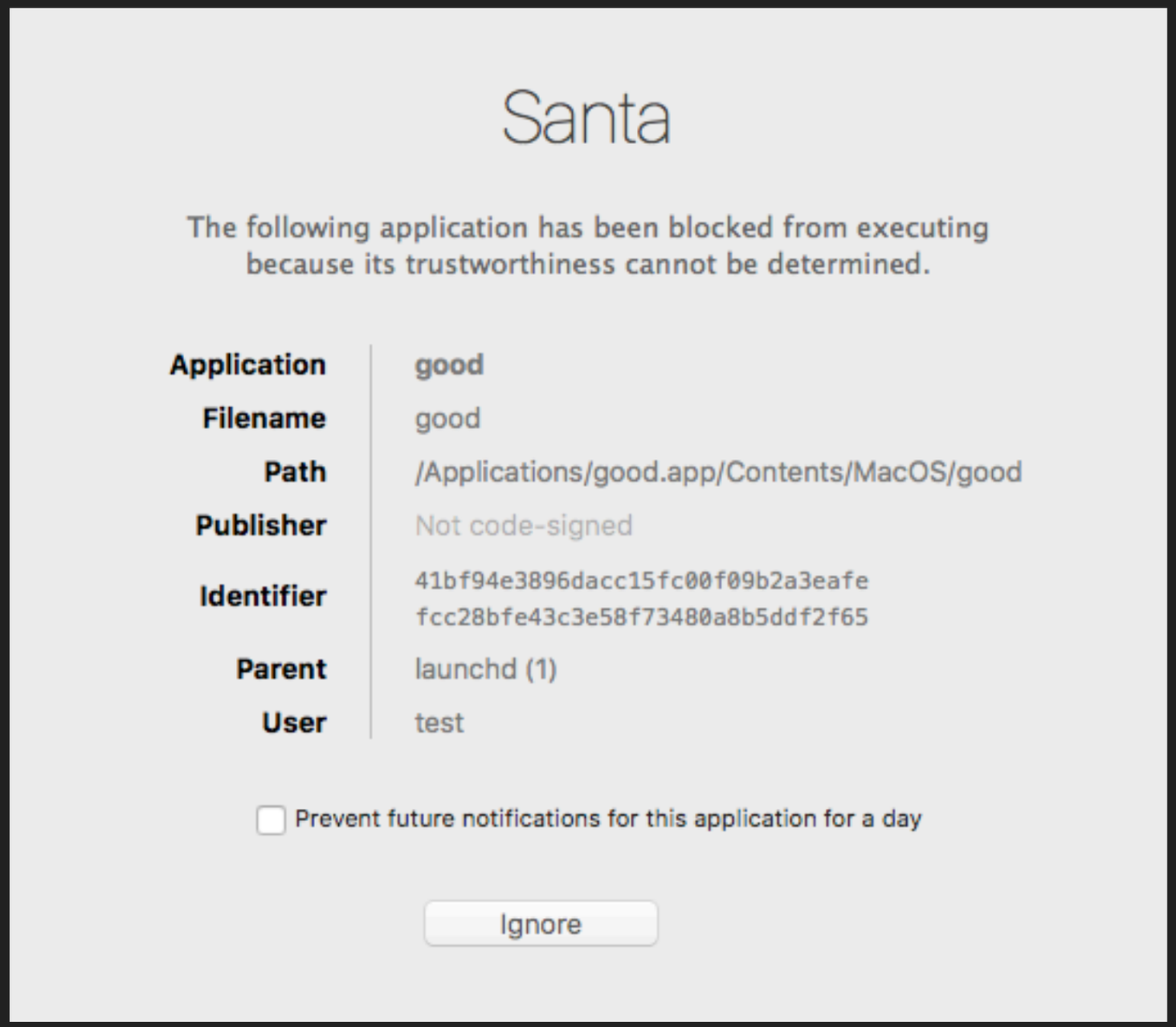
> Possible solution: Santa
  https://github.com/google/santa

> Use in lockdown mode to allow only whitelisted
  apps to run

> Modified apps will be blocked



Naughty or nice?



Naughty or nice?

> Pros:

  > Difficult to bypass

> Cons:

  > Whitelisting will keep you jumping with user requests!

  > Unrealistic for certain users (eg, developers)

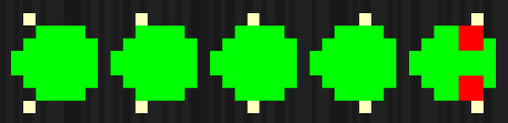
Thanks!

https://www.dropbox.com/s/yvs4iv91m773udd/Codesigning%20Mac.key?dl=0

Bonus points


.....Space Invaders.....10 points

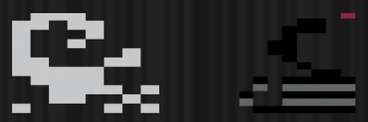

.....Blinky (PAC-MAN).....15 points


.....Centipede.....20 points


.....Pooka & Fygar (Dig-Dug).....50 points


.....Goomba (Super Mario).....75 points


.....Spider (Centipede).....100 points


.....snake & scorpion (Pitfall).....200 points


.....Coily (Q*bert).....500 points