

Picture XNU - Sockets Never Change

WarCon 2022

@jaakerblom

Picture XNU - Sockets Never Change

- This will be a talk about socket bugs
- Unlike my talk “Tales from the iOS/macOS Kernel Trenches” from earlier this year I won’t be going very deep into technical details
- References to technical resources will still be given for those interested

Agenda

- whoami (H3 Edition)
- A look back at (some) socket bugs
- Socket bug bingo

whoami

whoami



John Åkerblom
potmdehex



Follow

potmdehex

@potmdehex

github.com/potmdehex/homm...

github.com/potmdehex Joined January 2015

0 Following 69 Followers

Tweets

Tweets & replies

Media

Likes



potmdehex @potmdehex · Jul 25, 2016

HDE Mod 1.22, adds RMG integration for Polish H3 Gold owners and fixes reported crashes:

whoami

- Big enthusiast of the 1999 game Heroes of Might and Magic III, which I know has a big community here in Poland in the past and present
- Started my journey into reverse engineering largely through this game, creating enhancements but also finding and exploiting memory corruption bugs in it
- Even wrote a metasploit module for it, featured by Rapid7 in their August 2015 blog post “Hackers of Might and Magic”
- If you have metasploit installed, check homm3_h3m.rb (still 0day to this day also on the updated HD Edition sold on Steam - greetings Ubisoft)

whoami

- Background as kernel driver developer for the Windows platform in the antivirus industry starting in 2013, working with a focus on firewall solutions and other network related code
- Started working in offensive mobile security, initially with a focus on the Android kernel, in 2016
- Switched focus to the iOS kernel and subsequently worked on security and vulnerability research with a focus on this platform for 5 years 2017-2021
- Currently working as an independent security researcher - doing research for Apple's Security Bounty program amongst other things

A look back at socket bugs

A look back at socket bugs

- In 2017, me and my friend Fabiano Anemone had both recently started delving into Android kernel research
- We had a conversation about areas of the Android kernel that might be interesting to look into for finding bugs
- On the next slide I've written down what was said, as best I could from memory

A look back at socket bugs

- Fabiano: What about sockets?
- Me: Sockets? Man, it's been done. Look at Ping Pong root by Keen Team and all these socket CVEs
- Fabiano: Maybe there is still something there
- Me: If anybody is paying attention, sockets shouldn't still be full of bugs on any platform
- Narrator: Nobody was paying attention (except behind closed doors)



A look back at socket bugs

- What I said in that conversation hasn't aged well
- But I *did* look into sockets, making use of relevant experience from Windows
- It yielded results, some of which I'll talk about today

A look back at socket bug #1 (Android)

A look back at socket bug #1

- Checking commits to the Linux kernel at the time led to stumbling upon the following

```
From: Eric Dumazet <edumazet@google.com>
```

```
[ Upstream commit 657831fffc38e30092a2d5f03d385d710eb88b09a ]
```

```
syzkaller found a way to trigger double frees from ip_mc_drop_socket()
```

```
It turns out that leave a copy of parent mc_list at accept() time,  
which is very bad.
```

```
Very similar to commit 8b485ce69876 ("tcp: do not inherit  
fastopen_req from parent")
```

```
Initial report from Pray3r, completed by Andrey one.  
Thanks a lot to them !
```

A look back at socket bug #1

- This bug was initially not marked as a security issue and didn't have any CVE assigned
- It was however trivially exploitable for LPE on Linux/Android
- As is often typical in the Android world, there was a large patch gap during which the bug remained unpatched and still exploitable on Android devices

A look back at socket bug #1

- Eventually this issue was assigned CVE-2017-8890, for which a couple of related repos on Github have appeared
- @thinkycx has a repo titled “CVE-2017-8890” with exploits for this bug for Ubuntu as well as Android
- There is also a repo “cve-2017-8890-msf” by @7043mcgeep with code and details about the bug
- I’ve included some code from @7043mcgeep’s repo on the next slide to give a basic idea about how triggering the bug looked

A look back at socket bug #1

Pseudocode with explanations:

A machine running a kernel 4.10.15 and under is at risk if it is running the following

```
sockfd = socket(AF_INET, xx, IPPROTO_TCP);
setsockopt(sockfd, SOL_IP, MCAST_JOIN_GROUP, xxxx, xxxx);
bind(sockfd, xxxx, xxxx);
listen(sockfd, xxxx);
newsockfd = accept(sockfd, xxxx, xxxx);
close(newsockfd); // trigger release calls, handoff to RCU
sleep(5);         // wait for rcu to free()
close(sockfd);    // second free()
```


A look back at socket bug #2 (iOS)

A look back at socket bug #2


- In June 2018, I published a kernel r/w(/x) exploit for another socket bug, this time for iOS in the *multipath* subsystem, for a bug patched in iOS 11.4 - CVE-2018-4241
- This exploit - **multipath_kfree** - has remained relatively unknown, having been put aside by the jailbreak community (in part for needing the multipath entitlement - requiring a Apple Developer Certificate)
- This is a side note, but the community was convinced Ian Beer's expected upcoming exploit for the same bug *wouldn't* need the multipath entitlement
- To the community's surprise, it also needed the entitlement

slide from BaiJiuCon 2018

CVE-2018-4241

It's a heap overflow

- * We can overwrite an address that will be kfree'd, 100% reliably
- * The address we overwrite initially points to itself (simplified - it's actually a few bytes away)
- * We can choose how many bytes of the address to kfree we overwrite
- * **We have to pay Apple 1000 NOK to overwrite the address (Developer Certificate required)**

 **John Åkerblom** @jaakerblom · Jun 3, 2018
Replying to @s1guza @coolstarorg and 3 others
Ian's exploit is going to be using the same entitlement



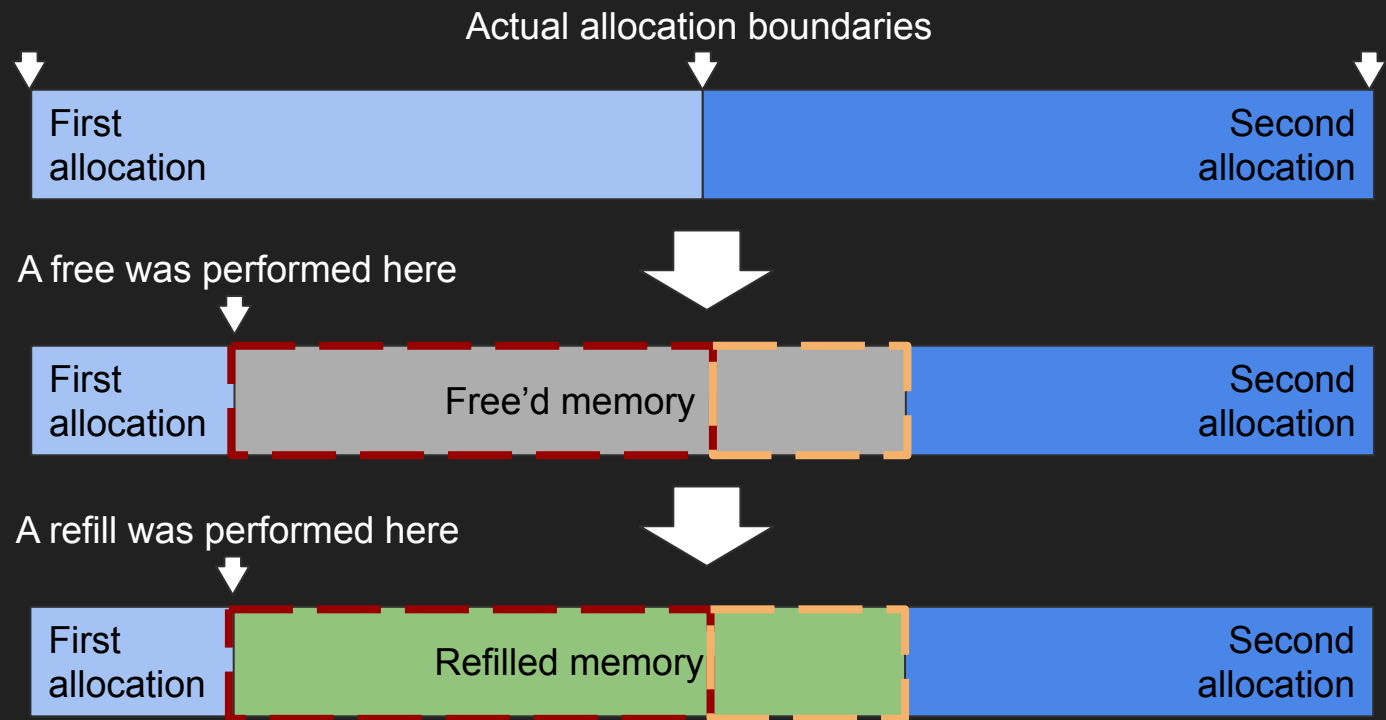
A look back at socket bug #2

- The `multipath_kfree` exploit seems to have been deemed of very little interest also by Apple
- `multipath_kfree` wasn't updated to be a grade A exploit in terms of cleanup/post execution reliability
- But it did use techniques that it arguably should have been in Apple's interest to take a look at
- It's a bit of a detour from sockets, but I've included a description of one technique in the following slides - the misaligned kfree technique

A look back at socket bug #2

- `multipath_kfree` used the technique of freeing and refilling memory at an address not actually aligned to an allocation boundary
- The ability to perform such kfree's was immensely useful
- It enabled type confusion of objects in ways otherwise not possible, overlapping members that normally wouldn't overlap
- It also allowed exploitation of bugs causing certain pointer corruptions (as an example, imagine a bug causing an existing pointer, which will be free'd, to be incremented by 8 - such frees didn't cause a panic)
- The next slide has some very basic graphics demonstrating the technique

A look back at socket bug #2



A look back at socket bug #2

- This technique remained unpatched until highlighted and explained in the June 2020 blog post “A survey of recent iOS kernel exploits” by Brandon Azad, which featured `multipath_kfree` (amongst many other exploits)
- When iOS 14 was released over 2 years after the release of `multipath_kfree` and not too long after Brandon Azad’s blog post, the technique was finally patched
- Of course, the blog post mentioned above may not have been the catalyst for this change - the point is it took a considerable amount of time and publicly posted research before this was addressed

A look back at socket bug #2

- As for the bug itself, it was a heap overflow deterministically granting a kfree primitive
- Triggering was done by calling connectx (multipath_kfree code below)

```
// Trigger overflow
connectx(sock, &sae, SAE_ASSOCID_ANY, 0, NULL, 0, NULL, NULL);

// We expect return value -1, errno 22 on success (but they don't guarantee it)

#ifdef MULTIPATH_ERRNO_CHECK
if (errno == 1)
{
    // Protip: Apple actually charges more than $100 for some regions (RIP 1000 SEK)
    *(int *)("You") = (int)"need to pay Apple $100 (add the multipath entitlement)";
}
```

connectx overflow struct

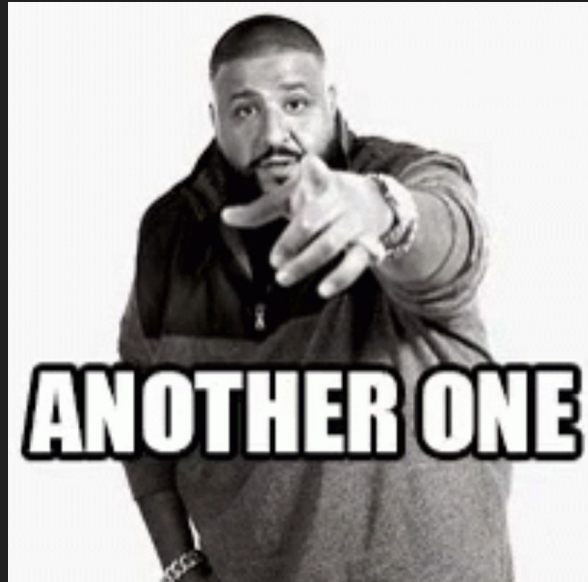
```
struct not_todescos_not_essers_ipc_object
{
    uint8_t zeroes[132-88];    // Unused by us
    uint32_t mpte_itfinfo_size; // If > 4, ->mpte_itfinfo free'd
    uint8_t nonzeros[168-136]; // Unused by us
    uint8_t nonzeros2[16];    // Unused by us
    uint64_t mpte_itfinfo;     // Address to free
};
```

- For details and a clear overview of the bug, I suggest anyone interested to look at the multipath_kfree Github repo and/or Ian Beer's Project Zero tracker issue and exploit

A look back at socket bug #3 (iOS)

A look back at socket bug #3

- This year, I published an iOS kernel r/w exploit for another socket bug - **multicast_bytecopy** for iOS 15



A look back at socket bug #3

- This bug, for iOS, is the second multicast bug in this presentation (the first of which was socket bug #1, for Linux/Android)
- Looking at Linux kernel commit logs isn't only useful for Android research - it's evidently good for iOS research too



A look back at socket bug #3

- For more information about this bug, CVE-2021-30937, I refer to the previously mentioned presentation "Tales from the iOS/macOS Kernel Trenches" for which the slides are public, as well as the Github repo "multicast_bytecopy"
- I may also expand on this bug in a future talk/write up
- Before moving on I would like to point out that this bug, which allows jailbreaking iOS 10 - iOS 15, is not just in the same subsystem but actually in the handling of the very same setsockopt option as socket bug #1 for Android from 2017 (code from earlier slide below)

```
setsockopt(sockfd, SOL_IP, MCAST_JOIN_GROUP, xxxx, xxxx);
```

A look back at setsockopt (iOS)

A look back at setsockopt

- In one of the many betas for iOS 14.5, the setsockopt attack vector using sockets was blocked in WebContent
- Bugs and primitives (e.g kernel heap sprays) available to attackers through this syscall were now finally no longer usable in typical 1 click attack chains
- However, attackers got one last hurrah for such bugs and primitives

A look back at setsockopt

- 14.5 was actually released with setsockopt **not** being blocked
- This was because Apple ended up **not** shipping the release based on the latest 14.5 beta
- Instead, the released 14.5 was based on an earlier beta where setsockopt was still allowed
- setsockopt exploitation thus remained business as usual on 14.5 for one more version until 14.5.1, while being highlighted by Apple as a probably pretty interesting attack/exploitation vector



A look back at setsockopt



A look back at setsockopt

- Researchers often discuss recently patched vulnerabilities
- Such discussions have revealed many researchers seem to have stopped looking into sockets for 1 click chains after 14.5.1
- Was 14.5.1 truly the end for sockets in WebContent?
- Sockets Never Change

A look back at socket bug #4 (iOS)

A look back at socket bug #4

- A very strong socket bug (which to my knowledge has yet to be mentioned anywhere publicly until this slide) that was **not** in setsockopt was still exploitable for kernel r/w from WebContent until iOS 15.1
- To be very clear, this one was in no way related to the bug fixed in iOS 15.2 exploited by me in the multicast_bytecopy exploit
- Details have been ~~censored~~ left out from this presentation
- I highly suggest anyone interested in socket bugs or good kernel bugs in general to take a look at iOS 15.1

Socket bug bingo

Socket bug bingo template

initially under the community's radar	grants free primitive easily	leads to LPE without needing other bugs	reliably and quickly exploitable	subsystem name starts with "multi"
------------------------------------------------------	---------------------------------------------	------------------------------------------------------------	-------------------------------------------------	-------------------------------------------------------

Socket bug bingo #1 (Linux/Android) BINGO!

initially under the community's radar	grants free primitive easily	leads to LPE without needing other bugs	reliably and quickly exploitable	subsystem name starts with "multi"
----------------------------------------------------------	-------------------------------------------------	----------------------------------------------------------------	-----------------------------------------------------	-----------------------------------------------------------

Socket bug bingo #2 (iOS) BINGO!

initially under the community's radar	grants free primitive easily	leads to LPE without needing other bugs	reliably and quickly exploitable	subsystem name starts with "multi"
----------------------------------------------------------	-------------------------------------------------	----------------------------------------------------------------	-----------------------------------------------------	-----------------------------------------------------------

Socket bug bingo #3 (iOS) BINGO!

initially under the community's radar	grants free primitive easily	leads to LPE without needing other bugs	reliably and quickly exploitable	subsystem name starts with "multi"
----------------------------------------------------------	-------------------------------------------------	----------------------------------------------------------------	-----------------------------------------------------	-----------------------------------------------------------

Socket bug bingo #4 (iOS)

Not in “multi” - no bingo

initially under the community's radar	grants free primitive easily	leads to LPE without needing other bugs	reliably and quickly exploitable	subsystem name starts with "multi"
----------------------------------------------------------	-------------------------------------------------	----------------------------------------------------------------	-----------------------------------------------------	---------------------------------------------

$\frac{3}{4}$ bingos achieved - not bad

Conclusion

Conclusion

- Sockets have had a lot of kernel bugs
- This presentation covered only a select few, with the many ones not covered including a fresh one patched in iOS 15.5 reported by Ned Williamson and unrestricted only last week

Fixed in iOS 15.5

<https://support.apple.com/en-us/HT213258>

Comment 4 by nedwill@google.com on Thu, Jun 16, 2022, 1:18 AM GMT+9

Labels: -Restrict-View-Commit

- The future of socket bugs looks pretty bright - I recommend more researchers to take a look into this area
- If the subsystem name starts with “multi”, it’s going to have bugs

Thanks to

- WarCon organizers and sponsors
- @_sinn3r (formerly) of Rapid7
- @krzywix
- Apple - not in a sarcastic way, I've only presented one side of things in this talk



Ian Fang @iantfang · Apr 6

Replying to [@jaakerblom](#)

“You ain’t seen nothing yet.” 😏



Pierre H. 🔥🌸 @pedantcoder · Jun 7

Replying to [@jaakerblom](#)

So how do you like iOS 16?