alfred - Bug #202

Multiple Master Syncing Robustness

01/16/2015 04:34 AM - Martin Weinelt

Status:	In Progress	**Start date:**	01/16/2015
Priority:	Normal	**Due date:**	
Assignee:	Hans-Werner Hilse	**% Done:**	10%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

When using Alfred on quite a lot of nodes the UDP Packages that Alfred sends to sync between masters is getting quite huge. It might therefore trigger fragmentation, and if only one fragment is lost the whole synchronization breaks.

The result is that when requesting data from a master it might result in no data or incomplete data.

Related issues:

Has duplicate alfred - Bug #232: alfred server loose data	**Rejected**	**02/19/2016**

History

#1 - 01/19/2015 02:02 PM - Simon Wunderlich

It is intended that only a complete number of sync packets are accepted to avoid "incomplete" data - therefore alfred only accepts a transaction if all data packets AND the final transaction package has been received.

The synchronization happens in regular interval, so if one transaction fails the next may succeed - unless you really have a lot of packet loss.

Can you please describe your scenario/problem in more detail? Then we might see if we can do something about that. However the general recommendation would be: make your underlying network at least somewhat reliable.

#2 - 01/19/2015 09:38 PM - Jan-Philipp Litza

We have the same problem. Setup is simply two nodes connected using a fastd VPN (MTU 1426) speaking batman-adv with MTU 1500 over that. This by the way is a totally usual setup for many Freifunk communities.

But we are using alfred with data that needs 3 UDP packets for syncing (I see some "Transaction finished" with sequence number 3 in a dump), meaning that at least 88* packets (probably more like >100) have to be transmitted without error. This means that a packet loss of around 1% almost certainly kills the synchronization. Even worse, because there is a VPN (and DSL) in between, this number even doubles because batman-adv has to fragment all the 1500 byte packets again, reducing the allowed packet loss to something like maybe 0.5%. This effectively makes multi-master unusable with big datasets over anything but plain ethernet where no fragmentation of 1500 byte packets happens and we have nearly no packet loss.

Also, these dataset sizes are real. Every Freifunk community using the gluon firmware generates several KBs of data per node, even though it is already gzip compressed. And we only have 160 nodes, there are bigger communities.

- 2 full UDP packets have a size of 2^16=65536 bytes, each of them being split up in roughly 44 IP packets 1500 bytes each, plus however big the third is, plus the "Transaction finished" packet.

#3 - 01/23/2015 04:41 PM - Simon Wunderlich

Thank you for these details, it is interesting to find out about your amounts of data and settings, which do have a big influence if the current method of synchronization is feasible or not.

What I wonder about is:

- What is your actual packet loss on ADSL lines? From my experience, <1% packet loss on ADSL is not that unusual ... * How often does the synchronization fails in your setup? * have you considered setting some Path MTU or similar to 1426 to work around the fragmentation issue?

We don't have any alternative for that sync mechanism currently, and if we need to change that we need to find out your problems first.

#4 - 01/24/2015 02:19 PM - Jan-Philipp Litza

Because we probably are more interested in running masters on multiple servers, I did testing between them instead of using an ADSL link (opposed to my last post, where I actually used the ADSL link because I didn't want to disturb our real setup). I used ping -f -c 1000 -s $size with varying values for $size. Results:

Loss in %	-s 1378	-s 1406	-s 1472	-s 65507
via fastd+batman-adv	0	None	3	87
via fastd	None	0	1	45
direct	None	None	0	14

"None" means not tested, because that particular size isn't special for that kind of link: I always tested the size fits through Ethernet link in one packet (1472=1500-20-8), the size that should fit through the used link in one packet (1378, 1406 and 1472 depending on the link) and 65507, which is the maximum size possible with fragmentation. I am unsure which is most interesting, but it may even be the 65507 one. HTH.

I will have to setup a testbed for counting the synchronization failures, so no numbers on that one yet.

And finally, I have no idea how I could set up a path MTU inside a layer 2 network. We use one large layer 2 network with all servers, nodes and clients in it, and for compatibility with the client devices we use an MTU of 1500. If I could lower this value for one particular host (so telling master1 to lower the MTU to master2 and vice versa), this would probably solve the problem - or at least make it much better. OTOH, if there isn't any way to do this on the system level, such an upper packet size bound could be implemented inside alfred so it splits its packets into more UDP packets. The default could even be 65507, but we could set it to 1378 in order to avoid IP fragmentation in our setup.

#5 - 02/18/2015 10:32 AM - Simon Wunderlich

- Assignee set to Simon Wunderlich

Thank you for your tests and sorry for the late reply.

It's interesting to see that you have different losses on these different layers. It seems batman-adv and fastd both add their own fancy fragmentation, which increases the probability to fail for large packets ... Which may not only affect alfred.

Anyway, there are probably two ways to address the problem:

- implement a packet size limitation inside alfred - this can probably be done within send.c/push_data(). It will limit your maximum size of data you can transmit with alfred, but as long as you don't send long messages that should be fine. This way we shift the fragmentation from batman-adv/fastd/IP to alfred only * set up some kind of MACVLAN device on top of your normal device you usually use for alfred, and decrease the MTU. That's yet another hack, but there does not seem to be any way to decrease the MTU on a per-socket basis, and Path MTU does only work with TCP afaik.

Let me know what you think - if you want to do the alfred hack and can't do yourself, I can supply a patch for you to test. If it works, we can add a general option to others.

Thanks

#6 - 03/30/2015 03:34 PM - Jan-Philipp Litza

I am now testing a setup with macvtap devices with an MTU of 1280 for all the (three) masters. It seems to work much more reliably, we'll see if it continues to do so. Thanks for the hint!

Only problem now: If a slave sends information larger than 1280 bytes, they probably get dropped due to the lower MTU on the masters. We're not there yet, but I think putting alfred behind a macvtap device on all the slaves is not really an option. So in the long run, maybe such a patch would be nice.

#7 - 05/11/2015 11:55 PM - Hans-Werner Hilse

- File 0001-Fix-maximum-payload-size-to-UDP-constraints.patch added

We are seeing problem with this in a "Freifunk" setup in Goettingen, too. Also, I found the "oldstable" (wheezy) Debian kernel to be quite unstable when it comes to receiving a bunch of large, fragmented UDP packets. 15 of those were enough to crash the kernel, though I'm lacking time and energy to debug an old 3.2 kernel.

That said, I found that the maximum payload size specified in alfred.h (0xFFFF) is too large in any case. In the setting alfred is supposed to run most of the time, 0xFFFF minus 8 bytes (UDP header) is probably the way to go. For all larger sizes, the kernel will refrain to even take the order. So the attached patch fixes that. Maybe it would be a good idea to make this constant a variable that can be set/overridden by a command line flag? I could send a patch for that, too.

#8 - 05/12/2015 05:53 PM - Sven Eckelmann

Hans, please send patches to the mailing list. See Contribute

But you are correct about -8.

#9 - 05/12/2015 05:56 PM - Sven Eckelmann

At least when ignoring IPv6 jumbograms

#10 - 05/13/2015 11:19 AM - Hans-Werner Hilse

OK, will do! I think I need to make that contribution a bit larger, since the packet split automatism will cope badly (as in: create a stack buffer overflow) for stored data of size > 65527 (which can happen since the MAX_PAYLOAD is only taken into account for sending packets, not receiving them). Probably, due to header structs, the buffer (buf) can overflow even now for a small corridor of data sizes (data size is checked, but the action taken if it is too large to fit is to send the currently accumulated data and then copy the new data into the buffer anyway - with the assumption that the buffer is then large enough). Practically, that overflow can't happen now since UDP packets larger than 65535 (overall) can't come in.

I'll need to think about this a bit.

Maybe using TCP for a) slave-to-master full request, and b) master-to-master sync is also an option. At least for a) in most cases a successful, near-time communication is asked for.

Going for a small MAX_PAYLOAD to avoid fragmentation will reduce possible maximum data item size (and will lead to aforementioned buffer overflow pretty soon). Other ways to do alfred-side fragmentation will make matters more complicated (transaction model needed). It's not a nice situation. I guess this discussion is also better taken to the ML, however. Thanks for getting back!

#11 - 08/14/2015 06:14 PM - Simon Wunderlich

any news how to proceed on this issue?

#12 - 01/29/2016 11:00 AM - Simon Wunderlich

- Status changed from New to Closed

Closing this ticket for inactivity.

Feel free to reopen if you still want to work on that.

#13 - 02/20/2016 09:34 AM - Sven Eckelmann

- Related to Bug #232: alfred server loose data added

#14 - 03/17/2016 10:49 AM - Dirk Thierbach

We are seeing the same problem in the Freifunk Frankfurt network. The Alfred master stores currently about 150 KBytes of data. When requesting this data using an Alfred client, the data gets split up in 3 UDP packets, the first two of which are about 64 KBytes large. These packets in turn are split into about 53 fragments, and at least one of those fragments invariably gets lost when trying a remote request to the Alfred master from the client. Which means no answer at all, as the fragements can't be reassembled and the packet is dropped completely.

There were also problems reported with running multiple Alfred masters in the network. I didn't investigate that, but this issue might also have been the reason for those problems.

Instead of fiddling around with maximum UDP packet size or maybe implementing a homegrown retransmit algorithm, I'd suggest using a TCP connection to request data at least from an Alfred master (and maybe also an Alfred client). This could also be used to synchronize data between masters (i.e., pulling data instead of pushing it).

Or is there any fundemental reason why Alfred should only use UDP? Pushing lots of small data packets from many clients to a master works fine with UDP, of course.

If nobody has objections, I can try to implement it.

#15 - 03/17/2016 11:13 AM - Hans-Werner Hilse

I'm now done with my exams. Feel free to give it a go, I think over the next days and maybe a few weeks, I will also have time to invest in this. I'll drop you a mail, maybe we can flesh things out together.

#16 - 03/17/2016 09:11 PM - Sven Eckelmann

- Assignee changed from Simon Wunderlich to Hans-Werner Hilse

- Status changed from Closed to New

#17 - 03/21/2016 01:48 PM - Sven Eckelmann

- % Done changed from 0 to 10

- Status changed from New to In Progress

For people which are not subscribed to the mailing list: Hans-Werner Hilse already posted his first RFC version of the patch. It would be nice when everyone interested in this change would review/test it.

https://patchwork.open-mesh.org/project/b.a.t.m.a.n./patch/1458551033-8734-2-git-send-email-hwhilse@gmail.com/

Thanks

#18 - 03/28/2016 07:49 AM - Sven Eckelmann

The updated RFC can be found under
https://patchwork.open-mesh.org/project/b.a.t.m.a.n./patch/1459103215-22444-1-git-send-email-hwhilse@gmail.com/

#19 - 04/24/2016 08:47 AM - Sven Eckelmann

Looks like there are open problems like assigning -1 to pointers (which may gets dereferenced) and casting request structures to uint8_t * and then back to the request structure type (pointer).

The user also reports crashes
https://lists.open-mesh.org/mailman3/hyperkitty/list/b.a.t.m.a.n@lists.open-mesh.org/message/DRA43GJA7MFHXPR4T2VXRXVWLZL3KQSZ/ and #232#note-12

#20 - 04/24/2016 08:47 AM - Sven Eckelmann

- Related to deleted (Bug #232: alfred server loose data)

#21 - 04/24/2016 08:47 AM - Sven Eckelmann

- Has duplicate Bug #232: alfred server loose data added

Files

0001-Fix-maximum-payload-size-to-UDP-constraints.patch	1.19 KB	05/11/2015	Hans-Werner Hilse