RuggedVPN Stable Firmware Release October 28th, 2016 - Version 2016100640/2016102400

This release brings very important improvements in regards of product performance, quality and stability. All existing customers should update to this release in a timely manner. We also recommend all customers still using Classic firmware to upgrade to this release now, as end of support for Classic firmware is nearing.

If you wish to upgrade from a Classic firmware, please first update the router to the last stable Classic firmware release (Version 2015081830/2015102900 released on November 27th 2015). Please note that upgrading your firmware from Classic to RuggedVPN requires a Viprinet Lifetime Maintenance license to be in place. For more information, please check https://www.viprinet.com/vlm.

It is possible to have Routers and Hubs running on Classic firmware connect to a device running RuggedVPN firmware. However, a compatibility mode will be used in this case, which limits performance and features. It is therefore not recommended to use such a setup in production permanently, but it is OK to have a Classic firmware device talk to a RuggedVPN firmware device while you are upgrading these devices. The Software VPN Client right now is still based on Classic Firmware, and therefore will connect in compatibility mode. A RuggedVPN-based VPN Client will become available soon.

The list below lists all new features and bug fixes compared to the third RuggedVPN firmware release (Version 2016080240/2016080800 released on August 11th 2016).

New features

- There are no new features in this release.

Bug fixes

- Channel shutdown/disconnect has not been clean for a while. We suspect that this could have lead to system crashes. In less-fatal situations it gave you SSL errors on disconnect, or had channels time out for up to 5 seconds instead of cleanly disconnecting right away.
- On certain products (310, 2620, 2030, 5000, 5010) the hardware crypto engine could crash during channel disconnects. We believe that this bug is the reason why customers running routers under high stress (Satellite connections, ships, vehicles with lots of reconnects) are seeing devices reboot, while others don't.
- Fixed stacking slaves having a calculated MTU < 1500 not getting that MTU used. This fixed slave channels on a 500 stacking slave (or anything that uses PPP) not getting used.

- Fixed automatic connect to the License Server, which did not work before (you had to manually connect to received updated licenses)
- Reverting back to not dialing directly for LTE modules, but instead modifying to profile to trigger call. This means that those customers complained about problems with private APNs prior to the current stable release now will have these problems again, but they can use the custom WWAN profile feature to work around. We had to revert this because with the new way of dialing Verizon in the US did not work at all anymore, and also problems have been reported from the UK.
- Fixed demo and project routers not being able to use stacking slave channels. Please note that you have to re-select the remote WAN module inside the channel after updating to this fixed firmware.
- Fixed popup message on demo routers to not say that the demo runs only 14 days, but correctly state that it is 90 days.
- Re-factored the way statistics about flow source and destinations host are managed. Before a DDoS attack using spoofed IP addresses could eat up all the RAM and a lot of performance. The system now no longer can be put out of service flooding it with spoofed IP addresses, and in generally also performs better in scenarios where a huge amount (1000+) of LAN devices are using a Viprinet router.
- In case of certain kind of DoS attacks, the routingcore thread potentially could have gone stuck, causing the router to reboot about 90 seconds.
- In case a DDoS attack is detected as likely, messages on this are logged, at the maximum once per minute.
- A bug in memory handling of IP packets has been corrected. This bug could cause both a small memory leak accumulate over time, but also could cause crashes under certain circumstances.
- Setting an IPv6 address as main LAN IP address (instead of adding the V6 address as an Alias) wasn't supported but not prevented, which could result in the router becoming unreachable from the network. It's no longer possible to set an IPv6 address as the main LAN interface address.
- The retransmission manager used for QoS classes having "Guaranteed delivery" enabled had a bug which first of all was leaking memory, but also could have resulted in half of the retransmissions not taking place. This means that a flow going through the tunnel could have seen packet loss even with Guaranteed delivery on. This could have dramatic consequences: TCP/IP header compression used builds on the guarantee that there never are lost packets. This means that if a flow used guaranteed delivery and then a channel had packet loss, the flow could get stuck.
- Fixed log output of TCP Sequence numbers (the numbers could be logged negative). Also lowered log level of very common TCP violations caused by broken scanners, no longer claiming that you are under attack.
- Make sure stuck stream downloads are aborted. This bug could have resulted in download tests doing in the webinterface eating 99% CPU if they get stuck.
- The routing core could get stuck on the receiving side of flows. Very rare, depending on timing. You could be running without a stuck routing core for weeks, and then suddenly have it getting stuck all the time.
- Individual high-load guaranteed delivery flow receivers could get stuck after a tunnel reconnect, and while doing so could eat up memory.

- The receive-side timeouts for non-guaranteed flows that make sure that we wait long enough (but not longer than) for all fragments to arrive is based on the assumed bonding latency of the combination of channels used. The filters used weren't fine-tuned, and not suitable at all for satellite connections. These non-tuned values could cause packets getting dropped receive-side (because the router would not wait long enough for all fragments of that packet to arrive). This could result in the receiver being at fault for not being able to use the full tunnel speed downloading from/through the tunnel. The filters have been completely rewritten.
- The "Average latency is ... ms, maximum allowed is ... ms, no alternatives, keeping channel." message is gone, as is the underlying idea: It does not make sense to keep the last channel with a latency higher than the maximum allowed one. If the last channel drops out for a moment, the tunnel layer will do the buffering.