Dwarfguard 1.0.0 performance tests.

Dwarfguard 1.0.0 follows performance tests of Dwarfguard 0.8.0 and MAMAS (preceding product branding) versions 0.7.0 and 0.6.0.
The results should be comparable when the new features are not enabled (especially DCL which may impact performance noticeably when a big number of devices is sending data) and when the new emulator version does not use data variations and BIG data profiles.

There are two different test types in Dwarfguard testing:

- stability tests ... runs for a number of hours. Tests stability under standard conditions and raises flags in case of memory leaks.
- benchmark tests ... to measure different HW / VMs for comparability and clue for sizing deployments.
  Proves Dwarfg deployment with particular sizing is able to handle the tested number of devices.

Next to the test results, HW specs for environments, test description and methodology is given in the document.

The Dwarfg 1.0.0 is intended to be used for up to 60000 devices.
The subject of early adopter version testing are stability and benchmark testings.
Find out more in test overview.

General Dwarfguard 1.0.0 performance testing verdict:

Stability basic tests	PASSED	2025-01-30	Test sets: Stab-16-2-H1
Stability medium test	not performed		
Stability max (30000) test	PASSED	2025-01-30	Test sets: Stab-16-2-H1
Stability long test	not performed		Test sets: Stab-4-672

Testing overview

As mentioned earlier (Testing summary), there are three types of performance tests. All are here because we need to measure:
- number of accepted and dropped requests (and resulting percentage) FOR ALL TESTS
- time for benchmark tests
- memory usage for stability tests
- CPU utilization (%) for stability tests
- dwarfgd log review to make detail search for warnings and errors (for stability tests)

Next to measurements there are a few important calculated metrics:
- ideal maximal throughput (benchmark test)
- typical maximal throughput (20% of ideal) (benchmark test)
- number of pushes per second (benchmark test)
- maximal recommended # of devices per deployment type (HW / VM specs)

Environment specs		CPU cores	CPU threads	RAM MiB	stability test	benchmark
Proxmox Linux container Intel Xeon E5 2.2GHz 4C HT	C1	1	1	512		
	C2	2	2	1024		
	C3	4	4	2048		
	C4	8	8	4096		
AWS instance	A1 (small)					
	A2 (medium)					
	A3 (big)					
Baremetal AMD E350@1.6 GHz 2C	H0	2	2	16384		
Baremetal Core i5 1.7GHz 4C HT	H1	4	8	32768		
Baremetal Core i7 2.7GHz 4C HT	H2	4	8	32768		
Baremetal Core i7 3.2GHz2 GHz 6C HT	H3	6	12	32768		

Tests specs	ID	SSL?	# of devices	push/work T	# of loops	# of minutes	Human-time	Notes
Stability	Stab-2-2	Yes	1000	2/2	N/A	120	2 hours	
	Stab-4-2	Yes	3000	4/2	N/A	120	2 hours	
	Stab-8-2	Yes	10000	8/2	N/A	120	2 hours	
	Stab-16-2	Yes	30000	16/2	N/A	120	2 hours	
	Stab-4-48	Yes	3000	4/2	N/A	2880	48 hours	
	Stab-8-48-top	Yes	40000	8/2	N/A	2880	48 hours	
	Stab-4-672	Yes	3000	4/2	N/A	40320	4 weeks	
Benchmark	Bench-6-SSL	Yes	1200	6/1,2,4,8	10	N/A		1.0: Varied data
	Bench-6-noSSL	No	1200	6/1,2,4,8	10	N/A		1.0: Varied data
	Bench-12-SSL	Yes	2400	12/1,2,4,8	10	N/A		1.0: Varied data
	Bench-12-noSSL	No	2400	12/1,2,4,8	10	N/A		1.0: Varied data
	Bench-24-SSL	Yes	4800	24/1,2,4,8	10	N/A		1.0: Varied data
	Bench-24-noSSL	No	4800	24/1,2,4,8	10	N/A		1.0: Varied data
	Bench-48-SSL	Yes	9600	48/1,2,4,8	10	N/A		1.0: Varied data
	Bench-48-noSSL	No	9600	48/1,2,4,8	10	N/A		1.0: Varied data

Test run map 1.0.0	C1	C2	C3	C4	A1	A2	A3	H0	H1	H2	H3
Stab-2-2											
Stab-4-2											
Stab-8-2											
Stab-16-2									Yes		
Stab-4-48											
Stab-8-48-top											
Stab-4-672											
Bench-6-SSL									Yes		
Bench-6-noSSL									Yes		
Bench-12-SSL									Yes		
Bench-12-noSSL									Yes		
Bench-24-SSL											
Bench-24-noSSL											
Bench-48-SSL											
Bench-48-noSSL											

Stability tests									
ID / Environment →			Stab-2-2 / C1	Stab-4-2 / C2	Stab-8-2 C3	Stab-16-2 C4	Stab-16-2 H1	Stab-4-48 H0	Stab-8-48-top H3	Stab-4-672 C2
Test specs		Devices	1000	3000	10000	30000	30000	3000	40000	3000
		Emulator threads	2	4	8	16	16	4	8	4
		Set: Time / minutes	120	120	120	120	120	2880	2880	40320
		Set loop time / sec	200	200	200	200	200	200	200	200
HW specs		CPU cores	1	2	4	8	8	2	12	2
		RAM / MiB	512	1024	2048	4096	32768	16384	32768	1024
Resources	Pre-test	Available MiB OS					30680.00			
		Used MiB OS					1358			
		dwarfgd RSS (MiB)					56			
	Post-test	Available MiB OS					30230			
		Used MiB OS					1807			
		mamasd RSS (MiB)					408			
		avg load 15					0.63			
		avg load 15 / cpu core	0.00	0.00	0.00	0.00	0.08	0.00	0.00	0.00
Results	Test numbers	Processesd loops					22			
		Total time					7382			
		Real Loop time/sec (1)	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	336	#DIV/0!	#DIV/0!	#DIV/0!
		Estimated data-pushes	36000	108000	360000	1080000	1080000	2592000	34560000	36288000
		Performed Pushes					930000			
		Estim. Avg reqs/sec	5.00	15.00	50.00	150.00	150.00	15.00	200.00	15.00
		Rough avg reqs/sec (1)	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	125.98	#DIV/0!	#DIV/0!	
	Errors	Data ERR/retries	0	0	0	0	0	0	2	
		push errors (%)	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	0.00	#DIV/0!	#DIV/0!	#DIV/0!
		push errors (1=100%)	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	0.000000000	#DIV/0!	#DIV/0!	#DIV/0!
		Crashes	0	0	0	0	0	0	0	
		Reboots	0	0	0	0	0	0	0	
		Log entries					4		0	
		Log analysis					1 mid-air, 3 delayed DCL			
	Calculations	RSS increase MiB	0	0	0	0	352	0	0	
		RSS increase %	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	628.57	#DIV/0!	#DIV/0!	
		MiB per device	0.000	0.000	0.000	0.000	0.012	0.000	0.000	
		RAM utilization %	0.00	0.00	0.00	0.00	4.14	0.00	0.00	
		Max safe # of devs	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	723888	#DIV/0!	#DIV/0!	
	Summary	RAM utilization note					perfect (low)			
		CPU utilization note					perfect (low)			
		Verdict					PASSED			TBD

Notes	
(1)	The results are not exact as the total time includes registration time. In reality, there are a little bit more requests per second and the loop time is a little shorter than that.

Log analysis explained	
Mid-air profile collision	(Warning) Configuration profile with default values for a firmware creation attempted more than once in parallel. As all of the Advantech router device types in this test share the same profile, this is perfectly possible to happen. No impact on the system or data.
Delayed DCL	The DCL (data change log) table contains data value changes for all the devices. Any DB synchronization errors results in a delay of the synchronization to a later time.

Results are # of processed device data pushes per second									
	Spec:	C1	C2	C3	C4	H0	H1	H2	H3
	CPU thr	1	2	4	8	2/2	4/8	4/8	6/12
	RAM	512	1024	2048	4096	16384	16384	32768	32768
	HW	Intel Xeon E5 2CPU 4/8 each				E-350	Core i5	Core i7	Core i7
	Arch	Server				L PWR	Mobile		Desktop
	GHz	2.2				1.6	1.7	2.7	3.2
Test	Handlers								
6-SSL	1						250.06		
	2						250.45		
	4						253.91		
	8						253.71		
6-noSSL	1						320.94		
	2						329.16		
	4						332.17		
	8						334.71		
12-SSL	1						343.76		
	2						346.07		
	4						347.26		
	8						347.45		
12-noSSL	1						379.64		
	2						386.51		
	4						389.76		
	8						394.93		
24-SSL	1								
	2								
	4								
	8								
24-noSSL	1								
	2								
	4								
	8								
48-SSL	1								
	2								
	4								
	8								
48-noSSL	1								
	2								
	4								
	8								
Ideal max devs		0	0	0	0	0	65016	0	
Safe max devs		0	0	0	0	0	52012	0	

Notes/colors explained	

Findings

The number of Dwarfguard handling threads has negligible effect on the throughput. The likely reason is that the processing ability for requests is much higher than the emulation ability even though the emlator HW was upgraded. Meaning that in the real situation, the system would be able to process higher number of requests per second.

The difference between SSL and noSSL test is visible in the 6-thread emulator run but not that much in 12-thread emulator run. Likely reason for that is that the 12-thread run was fully utilizating the emulator HW just by data generation while in the 6-thread run the machine had some reserves that were seen when the SSL was disabled.

Number of pushes/second in comparison with testing of 0.8 was significantly lower in the 12-thread run and it is another indication that the emulator HW was utilized. The emulator used for the test of version 0.8 was incapable of creating varied data and used smaller data pushes than the newer emulator version. Generating varied data is much more demanding than sending the same datafile over and over again, supporting the idea.

Comparison of some data with later versions

Notes:

Each minor version adds a lot of functionality but also brings optimizations. Comparable results (100%) are considered success and results within 101-120% of resources usage and 99-80% throughput are considered ok.

Testing methodics is updated (and testing tools improved) with every version making comparison very hard.

The measurements and comparison are based on the basic stability (Stab-16-2) and basic performance (originally Bench-4-SSL but that got replaced by Bench-6-SSL-2 in 0.8.0) test results.

Findings for 1.0.0

Computed memory consumtion per device stayed virtually the same although the 1.0.0 version manages slightly more data for each device - this is caused by optimizations mainly in the DB storage layer of the Daemon. Total memory footprint has risen but only slightly.

Throughput of the 6-thread SSL benchmark stayed virtually the same which is on one side an excellent result as since v1.0.0 the emulator sends varied data which require considerably more computations but on the other hand, as noted on the performance slide, the emulator is probably incapable of utilizing the Dwarfguard so in reality the peak push/second is much higher and in the next version we should use either weaker HW for Dwarfguard or much more threads for testing coupled with stronger emulator HW.

		RAM – RSS MiB	% of previous	KiB per device	% of previous	Pushes/sec	% of previous
0.6.0	C1	49	100	15	100	142	100
	C2	54	100	7	100	182	100
	C3	112	100	8	100	189	100
	C4						
	H1	61	100	8	100	193	100
0.7.0 - introduced DCL	C1	57	116	20	133	143	101
	C2	95	176	19	271	175	96
	C3	271	242	23	288	161	85
	C4	665	100	21	100	190	100
	H1	491	805	15	188	155	80
0.8.0	C1	57	100	13	65	126	88
	C2	75	79	10	53	237	135
	C3	186	69	14	61	284	176
	C4	415	62	12	57	324	171
	H1	395	80	12	80	237	153
	H0	88	100	13	100	97	100
1.0.0 - introduced varied data in emulator	H1	408	103	12	100	250	105