УДК 004.056.5 **DOI:10.30837/rt.2019.3.198.06**

О.О. КУЗНЕЦОВ, д-р техн. наук, В.А. ТИМЧЕНКО, К.Є. ЛИСИЦЬКИЙ,
М.Ю. РОДІНКО, М.С. ЛУЦЕНКО, К.Ю. ШЕХАНІН, А.О. КОЛГАТІН

ДОСЛІДЖЕННЯ ШВИДКОДІЇ ТА СТАТИСТИЧНОЇ БЕЗПЕКИ АЛГОРИТМІВ КРИПТОГРАФІЧНОГО ҐЕШУВАННЯ

Вступ

Стаття є продовженням попередніх робіт «Алгоритми криптографічного ґешування, які застосовуються в сучасних блокчейн-системах» та «Дослідження алгоритмів криптографічного ґешування, які застосовуються в сучасних блокчейн-системах».

В цій роботі проводяться порівняльні дослідження алгоритмів криптографічного ґешування, які застосовуються (або можуть застосовуватися) в сучасних децентралізованих блокчейн-системах. Зокрема досліджується швидкодія ґешування на різних десктопних системах, оцінюється кількість тактів обчислювальної системи на один байт (Cycles/byte), обсяг ґешованого повідомлення за одну секунду (MB/s) та кількість сформованих ґеш-кодів за секунду (KHash/s). Додатково проводяться дослідження швидкодії окремих криптографічних функцій ґешування на графічних обчислювачах. Для оцінки статистичної безпеки проводяться дослідження вихідних послідовностей криптографічних функцій ґешування при обробці ними надмірних вхідних даних (які сформовано за допомогою звичайного лічильника). Для порівняльних досліджень показників статистичної безпеки використовується методика NIST STS (Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications), яку рекомендовано Національним інститутом стандартів і технологій США для дослідження генераторів випадкових і псевдовипадкових чисел для криптографічних застосувань.

Порівняльні дослідження на обчислювальних десктоп-системах

Для проведення порівняльних досліджень застосовувалися еталонні та (за наявності) оптимізовані програмні реалізації алгоритмів ґешування інформації.

Дослідження проводилися на обчислювальних десктоп-системах:

- 64-розрядної обчислювальної платформи із застосуванням AMD Ryzen Threadripper 2970WX 24-Core Processor 3.0 GHz.;
- 64-розрядної обчислювальної платформи із застосуванням Intel Core i9-7980 2.60 GHz.

Дослідження швидкодії проводилися за трьома критеріями:

- кількість тактів обчислювальної системи на один байт (Cycles/byte);
- обсяг повідомлення за секунду, MB/s;
- кількість сформованих ґеш-кодів за секунду, KHash/s.

Зазначені критерії характеризують як абсолютні (другий та третій показники), так і питомі показники швидкодії (перший показник). Отримані дані містять результати тестування швидкості ґешування для різних довжин вхідних текстів. Зокрема на вхід кожного алгоритму подавалися повідомлення (послідовні значення лічильника) завдовжки від 2^0 до 2^{20} байтів.

У наступних таблицях наводяться зведені результати порівняльного аналізу швидкодії зазначених алгоритмів за кожною із застосованих обчислювальних десктоп-системах. Зокрема, у табл. 1 наведено зведені результати порівняльних досліджень швидкодії алгоритмів ґешування на 64-розрядної обчислювальної платформи із застосуванням AMD Ryzen Threadripper 2970WX 24-Core Processor 3.0 GHz. При цьому у якості вхідних даних подавався $2^0 = 1$, 2^{10} та 2^{20} байт даних. У табл. 2 наведено результати тестування швидкодії досліджуваних алгоритмів для 64-розрядної обчислювальної платформи Intel Core i9-7980 2.60 GHz (із довжинами вхідних даних 1, 1024 та 1048576) байтів.

Таблиця 1

Результати тестування швидкодії алгоритмів для вхідного блоку даних розміром 2^0 байт
для 64-розрядної обчислювальної платформи (AMD Ryzen Threadripper 2970WX 24-Core Processor 3.0 GHz)

Назва алгоритму	Вхідний текст довжини 2^0 байт			Вхідний текст довжини 2^{10} байт			Вхідний текст довжини 2^{20} байт		
	Cycles/byte	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**
ARGON2D	2625530499,00	0,000001	0,001128	2605872,83	0,001149	0,001122	2531,66	1,182493	0,001128
ARGON2I	2271644943,00	0,000001	0,001315	2232922,54	0,001335	0,001304	2180,86	1,367934	0,001305
BLAKE-224	769,39	3,89	3892,12	11,67	256,69	250,67	10,89	274,93	0,26
BLAKE-256	776,22	3,84	3842,20	11,77	254,39	248,42	10,98	272,71	0,26
BLAKE-384	1010,83	2,96	2963,25	8,18	366,63	358,04	7,15	418,59	0,40
BLAKE-512	1034,26	2,93	2931,19	8,21	364,85	356,30	7,17	417,26	0,40
BMW-224	693,03	4,32	4319,57	5,92	505,83	493,97	5,23	572,68	0,55
BMW-256	690,86	4,34	4336,18	5,89	508,28	496,36	5,20	575,82	0,55
BMW-384	729,17	4,10	4100,48	3,52	851,12	831,17	2,79	1073,26	1,02
BMW-512	729,74	4,10	4104,98	3,51	853,89	833,88	2,78	1076,57	1,03
CUBEHASH-224	5240,57	0,57	567,60	20,31	147,48	144,02	15,17	197,17	0,19
CUBEHASH-256	5286,08	0,57	566,63	20,54	145,53	142,12	15,35	195,16	0,19
CUBEHASH-384	5268,51	0,57	572,83	20,32	147,48	144,02	15,17	197,58	0,19
CUBEHASH-512	5249,72	0,57	570,21	20,37	147,02	143,58	15,19	197,10	0,19
DJB-2	$2{,}75\cdot10^{-5}$	$2{,}33\cdot10^{7}$	$2{,}33\cdot10^{10}$	$2{,}52\cdot10^{-5}$	$2{,}13\cdot10^{7}$	$2{,}08\cdot10^{7}$	$2{,}48\cdot10^{-5}$	$1{,}91\cdot10^{7}$	18181,8
ECHO-224	4755,88	0,63	629,14	27,72	108,21	105,68	24,56	121,91	0,12
ECHO-256	4696,26	0,64	637,73	27,40	109,31	106,74	24,32	123,17	0,12
ECHO-384	5861,37	0,51	511,03	52,56	57,01	55,67	45,92	65,23	0,06
ECHO-512	5865,36	0,51	510,73	51,13	58,53	57,16	45,43	65,90	0,06
ED2K	291,07	10,24	10239,00	4,02	747,38	729,86	3,75	798,00	0,76
EDONR-256	274,37	11,10	11097,22	3,89	765,94	747,99	3,66	816,65	0,78
EDONR-512	290,55	10,34	10338,95	2,17	1377,89	1345,60	1,91	1569,72	1,50
FUGUE-224	2187,69	1,37	1371,91	19,23	155,81	152,15	17,11	175,14	0,17
FUGUE-256	2144,03	1,40	1396,28	19,13	156,53	152,86	17,06	175,52	0,17
FUGUE-384	3244,03	0,92	923,73	31,62	98,87	96,55	26,40	113,49	0,11
FUGUE-512	4645,23	0,65	645,50	37,46	79,89	78,01	33,01	90,72	0,09
GOST34.11-94	3917,49	0,76	763,72	43,74	68,44	66,83	42,19	70,97	0,07
GROESTL-224	2155,56	1,40	1395,26	23,23	128,93	125,91	21,10	141,83	0,14
GROESTL-256	2132,20	1,40	1404,36	22,98	130,32	127,27	20,88	143,44	0,14
GROESTL-384	5217,52	0,57	574,17	31,30	95,62	93,38	26,67	111,92	0,11
GROESTL-512	5271,46	0,57	572,80	31,39	95,43	93,19	26,25	114,06	0,11

Назва алгоритму	Вхідний текст довжини 2^0 байт			Вхідний текст довжини 2^{10} байт			Вхідний текст довжини 2^{20} байт		
	Cycles/byte	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**
HAMSI-224	555,85	5,39	5391,69	32,69	91,67	89,53	32,51	92,07	0,09
HAMSI-256	561,81	5,33	5326,51	32,91	90,99	88,86	32,18	93,02	0,09
HAMSI-384	2035,20	1,48	1481,58	84,47	35,75	34,91	82,00	36,50	0,03
HAMSI-512	1996,16	1,50	1504,52	83,53	35,83	34,99	81,97	36,56	0,03
HAS160	363,10	8,25	8251,31	5,41	554,22	541,23	5,05	592,42	0,56
JH-224	4075,85	0,74	743,45	33,28	90,68	88,56	30,63	97,81	0,09
JH-256	4036,49	0,74	741,85	32,74	91,44	89,30	30,72	97,53	0,09
JH-384	4116,05	0,73	727,22	33,19	90,15	88,03	31,17	96,25	0,09
JH-512	4050,76	0,74	738,80	32,68	91,61	89,46	30,63	97,73	0,09
KECCAK-224	1481,99	2,02	2017,77	10,94	273,21	266,81	9,59	311,15	0,30
KECCAK-256	1467,41	2,04	2041,30	10,98	272,85	266,46	10,21	293,14	0,28
KECCAK-384	1487,79	2,01	2011,35	13,79	218,18	213,07	13,40	223,10	0,21
KECCAK-512	1492,17	2,01	2007,73	20,54	145,72	142,30	19,33	154,91	0,15
KUPYNA-256	1706,85	1,70	1697,96	19,45	157,04	153,36	17,83	173,03	0,17
KUPYNA-512	5519,54	0,54	541,42	34,29	89,77	87,66	28,72	106,81	0,10
LOSELOSE	$2{,}55{\cdot}10^{-5}$	$2{,}33{\cdot}10^{7}$	$2{,}33{\cdot}10^{10}$	$2{,}49{\cdot}10^{-5}$	1,94·107	$1{,}89{\cdot}10^{7}$	$2{,}47{\cdot}10^{-5}$	$1{,}97{\cdot}10^{7}$	18867,9
LUFFA-224	809,50	3,82	3816,75	12,21	240,55	234,92	11,35	265,06	0,25
LUFFA-256	764,97	3,91	3905,89	12,04	248,65	242,83	11,23	266,47	0,25
LUFFA-384	1607,08	1,86	1863,11	17,57	170,58	166,59	15,86	188,42	0,18
LUFFA-512	2124,64	1,41	1411,65	23,54	127,42	124,44	21,36	140,30	0,13
LYRA2REV2	20434,44	0,15	146,51	30,96	96,73	94,46	10,97	273,07	0,26
LYRA2RE	19600,40	0,15	152,81	30,22	99,20	96,88	10,90	275,22	0,26
MD4	264,12	11,35	11354,37	3,97	756,00	738,28	3,74	802,28	0,77
MD5	364,08	8,22	8224,13	5,63	532,00	519,53	5,26	568,03	0,54
PANAMA-256	1859,84	1,61	1610,96	3,42	873,81	853,33	1,58	1906,50	1,82
RIPEMD-160	856,08	3,51	3507,65	13,57	220,71	215,53	12,75	234,84	0,22
SCRYPT	1331505,00	0,00	3,33	883,21	3,41	3,33	23,40	128,19	0,12
SHA1	571,89	5,23	5227,72	8,80	340,12	332,14	8,25	362,95	0,35
SHA2-256	806,83	3,70	3704,69	12,60	237,66	232,09	11,80	253,71	0,24
SHA2-512	1045,45	2,88	2880,07	8,71	344,47	336,40	7,68	389,95	0,37
SHABAL-256	1687,56	1,77	1772,50	8,28	360,71	352,25	6,58	454,32	0,43
SHABAL-512	1677,05	1,79	1785,93	8,32	359,96	351,53	6,55	458,49	0,44
SHAVITE-224	1021,71	2,91	2909,96	16,86	176,32	172,19	15,89	188,29	0,18
SHAVITE-256	1014,74	2,97	2966,94	16,60	180,42	176,19	15,62	191,73	0,18

Назва алгоритму	Вхідний текст довжини 2^0 байт			Вхідний текст довжини 2^{10} байт			Вхідний текст довжини 2^{20} байт		
	Cycles/byte	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**
SHAVITE-384	3149,33	0,95	952,58	27,45	109,06	106,50	24,42	122,78	0,12
SHAVITE-512	3112,57	0,96	961,91	27,21	110,04	107,46	24,17	123,90	0,12
SIMD-224	2743,21	1,11	1114,87	22,27	134,05	130,91	20,97	142,84	0,14
SIMD-256	2677,69	1,12	1118,79	22,30	134,35	131,20	20,96	142,82	0,14
SIMD-384	6141,45	0,49	490,09	27,11	110,73	108,13	24,09	124,24	0,12
SIMD-512	6162,56	0,49	485,97	27,04	110,81	108,21	24,00	124,77	0,12
SKEIN-224	582,96	5,15	5151,19	4,57	657,41	642,01	4,22	707,54	0,67
SKEIN-256	591,68	5,06	5062,89	4,59	654,54	639,20	4,25	703,74	0,67
SKEIN-384	598,34	5,03	5026,73	4,69	638,60	623,63	4,34	690,31	0,66
SKEIN-512	583,79	5,14	5136,55	4,53	661,15	645,65	4,21	710,90	0,68
SNEFRU-256	6740,30	0,44	441,18	108,35	27,63	26,99	105,08	28,49	0,03
STREEBOG-256	5278,11	0,57	566,96	35,10	85,31	83,31	29,98	100,02	0,10
STREEBOG-512	5260,38	0,57	569,58	35,01	85,52	83,52	29,86	100,28	0,10
TIGER	319,36	9,37	9371,49	4,25	705,16	688,63	3,95	758,19	0,72
WHIRLPOOL	1157,65	2,59	2586,46	18,05	166,18	162,28	16,93	177,24	0,17
X11	38652,22	0,08	77,50	45,04	66,50	64,95	7,28	411,37	0,39
X12	45946,76	0,07	65,17	52,14	57,40	56,06	7,26	412,83	0,39
X13	52768,44	0,06	56,79	58,66	50,81	49,62	7,22	414,62	0,40
X14	55180,53	0,05	54,29	61,13	48,97	47,83	7,25	413,15	0,39

Таблиця 2

Результати тестування швидкодії алгоритмів ґешування для 64 розрядної обчислювальної платформи (Intel Core i9-7980 2.60 GHz)

Назва алгоритму	Вхідний текст довжини 2^0 байт			Вхідний текст довжини 2^{10} байт			Вхідний текст довжини 2^{20} байт		
	Cycles/byte	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**
ARGON2D	1995096226,30	0,000001	0,001308	1904191,27	0,001361	0,001329	1866,23	1,386180	0,001322
ARGON2I	1983073690,80	0,000001	0,001329	1905170,88	0,001360	0,001328	1880,48	1,378617	0,001315
BLAKE-224	662,58	3,94	3936,10	10,25	259,23	253,15	9,20	281,65	0,27
BLAKE-256	648,36	3,99	3991,69	9,80	264,93	258,72	9,22	281,12	0,27
BLAKE-384	868,40	3,03	3025,58	6,63	392,14	382,95	5,81	446,39	0,43
BLAKE-512	846,69	3,06	3059,12	6,74	385,36	376,33	5,92	438,00	0,42
BMW-224	646,35	3,97	3966,77	5,57	462,34	451,50	4,96	524,29	0,50
BMW-256	649,68	3,92	3916,69	5,53	465,62	454,71	4,92	527,19	0,50
BMW-384	626,91	4,13	4131,02	3,01	862,32	842,11	2,39	1073,26	1,02
BMW-512	638,02	4,07	4071,35	3,05	851,12	831,17	2,43	1064,54	1,02
CUBEHASH-224	5272,54	0,49	489,42	20,20	127,94	124,94	14,94	173,40	0,17
CUBEHASH-256	5220,66	0,50	496,62	20,13	128,75	125,74	14,91	173,84	0,17
CUBEHASH-384	5150,19	0,50	503,98	20,71	125,16	122,22	14,69	176,59	0,17
CUBEHASH-512	5244,95	0,49	494,42	20,22	128,20	125,20	14,95	173,15	0,17
DJB-2	$4{,}96\cdot10^{-5}$	$1{,}83\cdot10^{7}$	$1{,}83\cdot10^{10}$	$1{,}53\cdot10^{-5}$	$2{,}27\cdot10^{7}$	$2{,}22\cdot10^{7}$	$1{,}41\cdot10^{-5}$	$2{,}23\cdot10^{7}$	21276,6
ECHO-224	4138,87	0,63	625,13	24,11	107,46	104,94	21,41	121,04	0,12
ECHO-256	4123,82	0,63	628,47	24,05	107,86	105,33	21,35	121,46	0,12
ECHO-384	5106,91	0,51	508,12	46,74	55,45	54,15	41,32	62,69	0,06
ECHO-512	5158,78	0,50	504,41	45,03	57,55	56,21	39,91	64,77	0,06
ED2K	240,10	10,84	10839,12	3,32	780,19	761,90	3,11	835,52	0,80
EDONR-256	265,47	10,15	10153,73	3,69	699,52	683,12	3,44	756,00	0,72
EDONR-512	244,48	10,61	10609,90	1,86	1387,01	1354,50	1,64	1572,08	1,50
FUGUE-224	1925,97	1,35	1348,29	17,63	147,21	143,76	15,62	165,94	0,16
FUGUE-256	1913,48	1,36	1355,17	17,58	147,46	144,00	15,69	165,16	0,16
FUGUE-384	3175,04	0,83	829,77	28,14	92,07	89,91	25,29	102,47	0,10
FUGUE-512	4344,18	0,60	596,73	35,86	72,31	70,62	31,25	82,89	0,08
GOST34.11-94	3311,63	0,79	785,16	36,76	70,44	68,79	35,85	72,40	0,07
GROESTL-224	1879,30	1,38	1379,58	21,74	118,46	115,68	19,22	134,92	0,13
GROESTL-256	1881,80	1,38	1377,75	20,86	124,28	121,37	19,22	136,11	0,13
GROESTL-384	4940,77	0,52	523,82	30,30	85,65	83,64	25,25	102,58	0,10
GROESTL-512	4854,19	0,53	533,60	33,12	85,24	83,25	26,06	102,69	0,10
HAMSI-224	543,88	4,78	4781,69	30,21	85,33	83,33	29,13	88,95	0,08

Назва алгоритму	Вхідний текст довжини 2^0 байт			Вхідний текст довжини 2^{10} байт			Вхідний текст довжини 2^{20} байт		
	Cycles/byte	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**
HAMSI-256	536,07	4,86	4857,22	29,49	87,94	85,88	28,94	89,66	0,09
HAMSI-384	2090,89	1,24	1238,35	88,41	29,30	28,61	85,88	30,20	0,03
HAMSI-512	2106,47	1,24	1235,01	87,49	29,31	28,62	85,98	30,33	0,03
HAS160	349,00	7,46	7462,11	5,03	514,51	502,45	4,72	548,99	0,52
JH-224	3640,96	0,70	701,32	30,40	83,81	81,84	28,42	91,23	0,09
JH-256	3648,43	0,71	710,37	29,69	87,48	85,43	27,85	93,26	0,09
JH-384	3748,59	0,69	692,15	30,37	85,42	83,42	28,46	91,12	0,09
JH-512	3706,28	0,70	701,99	30,35	85,60	83,59	27,86	92,64	0,09
KECCAK-224	1352,24	1,92	1916,78	9,45	274,07	267,64	8,36	310,51	0,30
KECCAK-256	1307,78	1,97	1974,12	9,64	268,87	262,56	8,96	289,50	0,28
KECCAK-384	1319,91	1,96	1964,84	12,08	214,70	209,66	11,82	217,68	0,21
KECCAK-512	1269,54	2,04	2041,02	17,41	149,28	145,79	16,40	158,11	0,15
KUPYNA-256	1660,20	1,54	1543,95	19,24	134,85	131,69	17,46	148,59	0,14
KUPYNA-512	5247,62	0,50	500,51	31,92	81,63	79,72	26,86	96,91	0,09
LOSELOSE	$4{,}96\cdot10^{-5}$	$1{,}49\cdot10^{7}$	$1{,}49\cdot10^{10}$	$1{,}41\cdot10^{-5}$	$2{,}55\cdot10^{7}$	$2{,}49\cdot10^{7}$	$1{,}41\cdot10^{-5}$	$2{,}38\cdot10^{7}$	22727,3
LUFFA-224	760,28	3,44	3443,26	11,87	218,09	212,98	11,25	230,56	0,22
LUFFA-256	777,22	3,35	3351,58	12,12	213,82	208,81	11,46	226,62	0,22
LUFFA-384	1622,45	1,59	1590,97	18,07	143,54	140,18	16,34	158,66	0,15
LUFFA-512	2077,64	1,25	1253,18	23,46	110,97	108,37	20,70	125,23	0,12
LYRA2REV2	19363,25	0,13	133,94	28,52	90,86	88,73	9,49	276,67	0,26
LYRA2RE	16422,12	0,16	157,76	25,67	100,92	98,56	9,44	274,50	0,26
MD4	221,46	11,76	11761,93	3,31	783,69	765,32	3,10	836,19	0,80
MD5	317,29	8,21	8206,10	4,91	527,45	515,09	4,62	561,04	0,54
PANAMA-256	1729,69	1,51	1511,79	3,23	806,60	787,69	1,49	1747,63	1,67
RIPEMD-160	743,69	3,49	3489,09	11,85	218,04	212,93	11,15	232,40	0,22
SCRYPT	2463209,70	0,00	1,11	722,61	1,80	1,75	20,56	125,73	0,12
SHA1	505,14	5,14	5136,05	7,81	331,83	324,05	7,41	349,76	0,33
SHA2-256	848,68	3,06	3060,37	13,46	192,54	188,03	12,68	204,88	0,20
SHA2-512	945,51	2,74	2741,59	7,94	327,37	319,70	7,02	369,87	0,35
SHABAL-256	1235,41	2,10	2097,95	6,15	420,61	410,75	4,94	524,55	0,50
SHABAL-512	1274,73	2,04	2039,99	6,31	416,60	406,83	5,01	507,78	0,48
SHAVITE-224	876,95	2,94	2942,55	14,19	180,07	175,85	13,46	191,91	0,18
SHAVITE-256	883,77	2,92	2915,87	14,33	178,63	174,45	13,53	189,93	0,18
SHAVITE-384	2805,29	0,92	922,12	24,41	105,95	103,47	21,67	119,52	0,11

Назва алгоритму	Вхідний текст довжини 2^0 байт			Вхідний текст довжини 2^{10} байт			Вхідний текст довжини 2^{20} байт		
	Cycles/byte	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**	**Cycles/byte**	**MB/s**	**KHash/s**
SHAVITE-512	2788,42	0,93	925,64	23,77	106,31	103,82	21,04	123,14	0,12
SIMD-224	2319,64	1,10	1095,10	19,98	130,05	127,00	18,78	138,13	0,13
SIMD-256	2335,71	1,12	1115,35	19,32	134,62	131,47	18,14	142,92	0,14
SIMD-384	5564,95	0,47	465,67	24,52	105,58	103,10	21,74	119,05	0,11
SIMD-512	5457,79	0,47	469,52	23,88	108,45	105,91	21,66	119,13	0,11
SKEIN-224	532,88	4,87	4870,75	4,12	629,78	615,02	3,88	669,16	0,64
SKEIN-256	526,25	4,94	4935,40	4,02	643,69	628,61	3,78	684,45	0,65
SKEIN-384	537,25	4,83	4833,48	4,15	627,14	612,44	3,90	666,19	0,64
SKEIN-512	525,12	4,93	4932,85	4,02	645,67	630,54	3,78	685,79	0,65
SNEFRU-256	5557,12	0,47	466,27	89,21	29,02	28,34	86,62	29,95	0,03
STREEBOG-256	4580,11	0,57	566,73	30,40	85,40	83,40	25,96	99,85	0,10
STREEBOG-512	4642,42	0,56	558,79	31,59	82,20	80,28	26,84	96,91	0,09
TIGER	240,20	10,87	10871,71	3,51	739,48	722,14	3,31	787,81	0,75
WHIRLPOOL	975,33	2,66	2659,88	15,57	166,60	162,69	14,62	177,27	0,17
X11	35972,07	0,07	71,73	40,95	63,76	62,26	5,91	439,47	0,42
X12	43318,45	0,06	59,35	48,72	53,19	51,94	5,98	433,12	0,41
X13	49509,65	0,05	52,36	54,36	47,66	46,54	5,93	437,09	0,42
X14	51112,37	0,05	50,75	55,84	46,40	45,31	5,93	437,64	0,42

Аналіз та дослідження показують, що найбільш швидкими є алгоритми ґешування DJB-2 та LOSELOSE. Але, як слідує із їхньої специфікації [1, 2], це не криптографічні алгоритми. Ці функції ґешування обчислюють звичайну контрольну суму і пошук прообразів для таких перетворень є тривіальним.

Далі за швидкістю перетворень йдуть алгоритми MD4, EDONR-256, EDONR-512, ED2K та інші криптографічні перетворення, стійкість яких є на сьогоднішній день не є задовільною [3 – 7].

Більшість криптографічних функцій мають порівняні показники швидкості формування ґеш-кодів [8 – 13], серед яких і алгоритм криптографічного ґешування Купина (національний стандарт України).

Найповільнішими алгоритмами ґешування виявилися ґеш-функції сімейства «X» та алгоритми ARGON2D, ARGON2I та SCRYPT [14-16].

Порівняльні дослідження швидкодії алгоритмів ґешування на графічних обчислювальних системах

Для проведення порівняльних досліджень швидкодії алгоритмів ґешування на графічних обчислювальних системах було обрано наступні апаратні засоби:

- Geforce 740M 2GB;
- Geforce GTX1050ti 4GB;
- Rx580 Aorus 4GB;
- Rx580 Sapphire Pulse 8GB;
- Sapphire Vega 56 8GB.

Проводилися вибіркові дослідження для наступних алгоритмів:

- ГОСТ 34.311;
- СТРИБОГ256;
- СТРИБОГ512;
- KECCAK 256;
- KECCAK 512;
- SHA2 256;
- SHA2 512;
- RIPEMD160;
- Blake2b;
- Wirlpool.

Для проведення досліджень застосовувалося програмне забезпечення HashCat [8]. HashCat – утиліта, яка надає можливість відновлення пароля. Найбільш активно вона використовується для відновлення WPA / WPA2 паролів, а також ключів від зашифрованих офісних документів. З 2015 р. поширюється з відкритим вихідним кодом під ліцензією MIT. Утиліта дозволяє використовувати будь-які пристрої, що реалізують стандарт OpenCL (OpenCL забезпечує паралелізм на рівні інструкцій і на рівні даних і є здійсненням техніки GPGPU. OpenCL є повністю відкритим стандартом, його використання не обкладається ліцензійними відрахуваннями).

Отримані результати тестування наведено у табл. 3 та 4.

В табл. 3 наведено результати Бенчмарку, тобто оцінки швидкості ґешування (кількість сформованих ґеш-кодів за секунду) шляхом послідовного ґешування набору даних.

В табл. 4 наведено питомі показники складності, а саме швидкість ґешування (кількість сформованих ґеш-кодів за секунду), яка приходиться на одне обчислювальне ядро OpenCL застосованого графічного обчислювача.

Таблиця 3

Результат Бенчмарка на графічних обчислювальних системах із застосуванням програмного забезпечення HashCat, Кілоґеш/с

	Geforce 740M 2GB	Geforce GTX1050ti 4GB	Rx580 Aorus 4GB	Rx580 Sapphire Pulse 8GB	Sapphire Vega 56 8GB
ГОСТ 34.311	10442,9	65337,9	89450	91932,3	233100
СТРИБОГ256	3512.6	13613,5	56751,7	55162,9	99485
СТРИБОГ512	3518,4	13569,5	55207,8	56635,9	99641,5
KECCAK 256	38149,7	260400	320400	327800	529600
KECCAK 512	38285	262400	326400	334400	538700
SHA2 256	133400	889600	1700900	1755900	3088100
SHA2 512	38794,4	297300	405600	421600	710100
RIPEMD160	177000	1383100	2356200	2437700	4129100
Blake2b	96515,9	585100	1103300	1132800	1738000
Wirlpool	12516,8	64773,3	324500	333200	591400

На рис. 1 для наочності наведено діаграму швидкостей ґешування із табл. 10. Як бачимо, найшвидшим є алгоритм RIPEMD160, далі слідує алгоритм SHA2, Blake2b та інші.

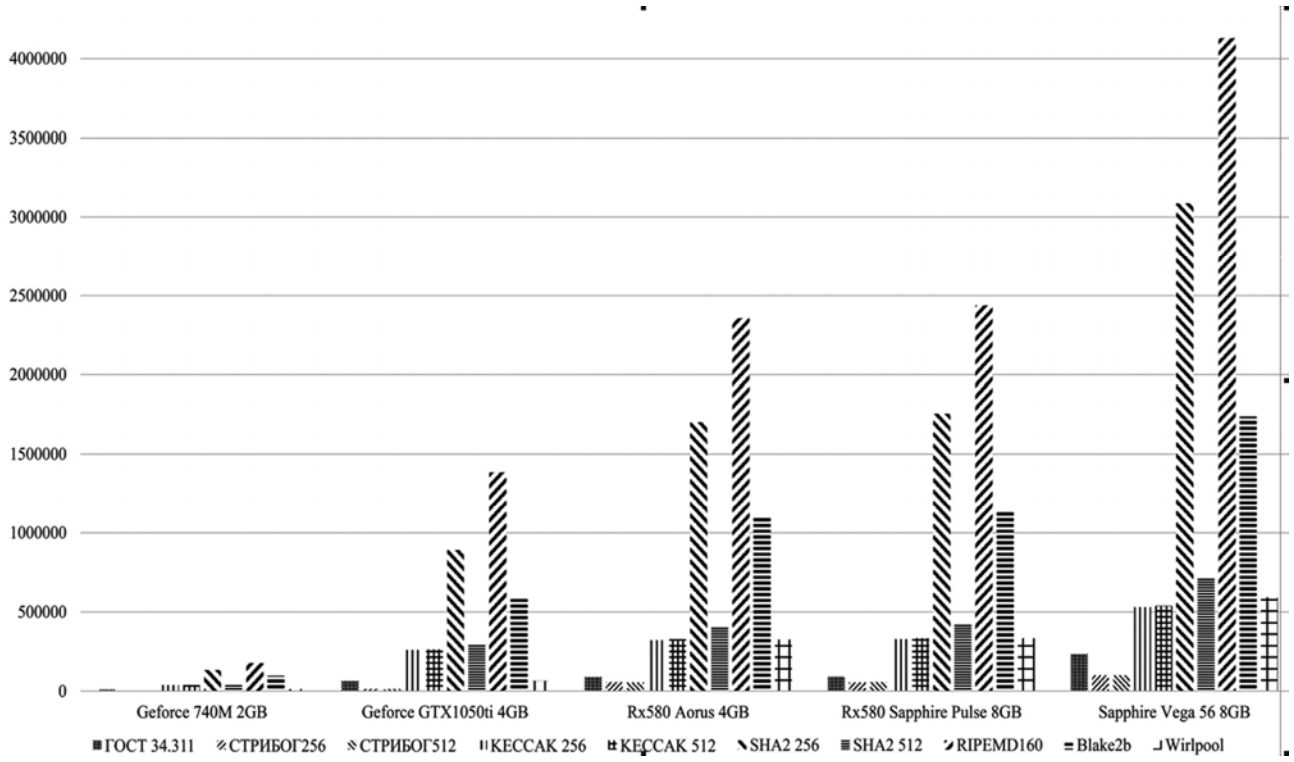

Рис. 1. Порівняння швидкодії алгоритмів ґешування на графічних обчислювачах

На рис. 2 наведено діаграми питомої швидкості, тобто швидкодія ґешування, що приходиться на одне обчислювальне ядро графічного обчислювача. Як бачимо, ранжування алгоритмів за швидкодією майже таке саме (RIPEMD160, SHA2, Blake2b та інш.), але власні значення швидкості змінилися. Тобто кожен графічний пристрій має різну кількість обчислювальних ядер і найбільш швидкими за критерієм питомої швидкодії виглядають обчислювачі Geforce.

Таблиця 4

Швидкість ґешування (кількість сформованих ґеш-кодів за секунду), яка приходиться на одне обчислювальне ядро OpenCL застосованого графічного обчислювача, Кілоґеш/с

	Geforce 740M 2GB	Geforce GTX1050ti 4GB	Rx580 Aorus 4GB	Rx580 Sapphire Pulse 8GB	Sapphire Vega 56 8GB
ГОСТ 34.311	5221,45	10889,65	2484,7	2553,6	4162,5
СТРИБОГ256	2706.3	2268,9	1576,4	1532,3	1776,5
СТРИБОГ512	1759,2	2261,5	1533,5	1573,2	1779,3
KECCAK 256	19074,8	43400	8900	9105,5	9457,1
KECCAK 512	19142,5	43733,3	9066,6	9288,8	9619,6
SHA2 256	66700	148266,6	47247,2	48775	55144,6
SHA2 512	19397,2	49550	11266,6	11711,1	12680,3
RIPEMD160	88500	230516,6	65450	67713,8	73733,9
Blake2b	48257,9	97516,6	30647,2	31466,6	31035,7
Wirlpool	6258,4	10795,5	9013,888889	9255,5	10560,7

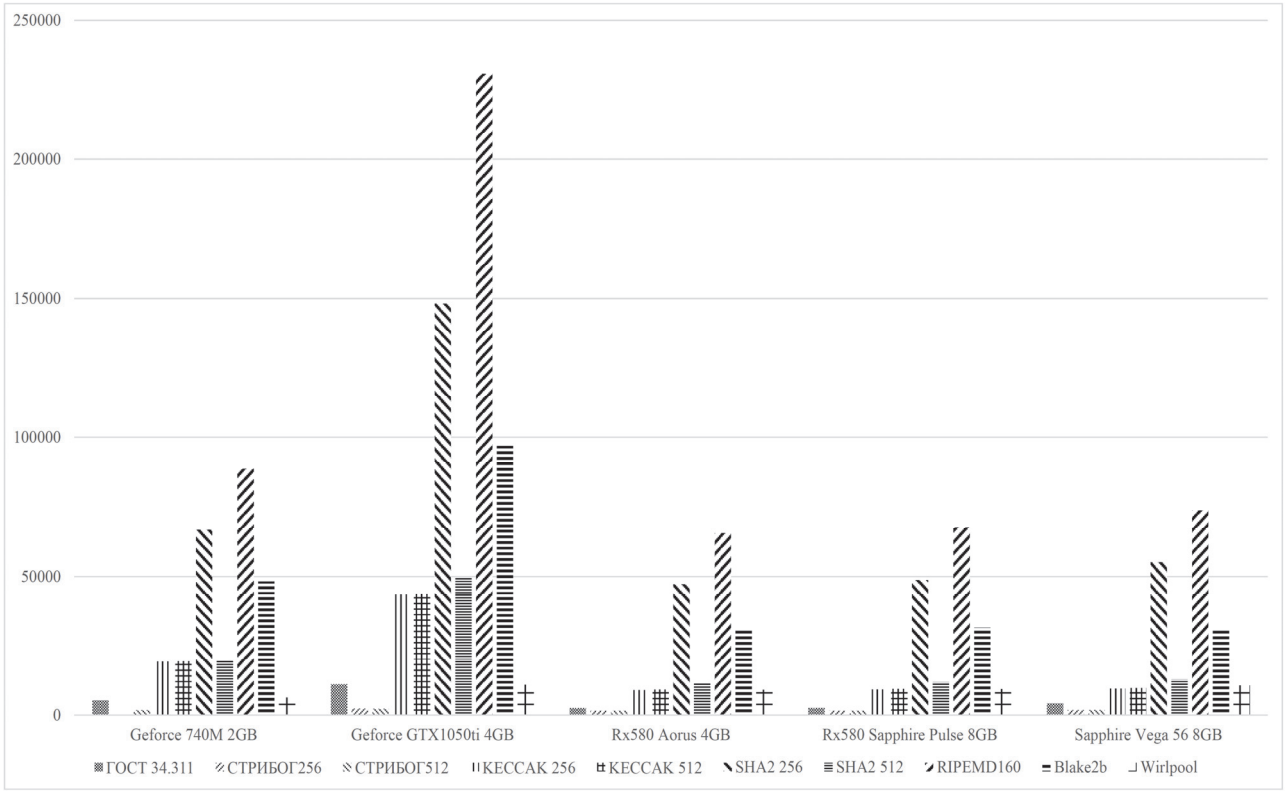

Рис. 2. Порівняння швидкодії алгоритмів ґешування на графічних обчислювачах

Таким чином, проведені дослідження показують, що різні за своєю структурою та математичними перетвореннями криптографічні функції ґешування дають різне прискорення на обчислювальних системах. Найбільш привабливими є графічні або спеціалізовані обчислювальні пристрої.

Методика та результати досліджень статистичної безпеки

Для проведення досліджень різних алгоритмів ґешування за критеріями статистичної безпеки було застосовано пакет статистичного тестування NIST STS, який рекомендований Національним інститутом стандартів і технологій США [18, 19]. Методику статистичного тестування та алгоритм обробки отриманих результатів наведено у роботах [20, 21].

Пакет статистичного тестування NIST STS був розроблений в ході проведення конкурсу AES для дослідження генераторів випадкових або псевдовипадкових чисел і є найбільш поширеним інструментом оцінки статистичної безпеки криптографічних примітивів. Використання даного пакету дозволяє оцінити, наскільки близько досліджувані криптоалгоритми апроксимують генератори «випадкових» послідовностей, тобто з високою ймовірністю стверджувати чи є генерована послідовність статистично безпечною. Порядок тестування окремої двійкової послідовності *S* має наступний вид :

- висувається нульова гіпотеза H_0 – припущення про те, що дана двійкова послідовність *S* є випадковою;
- за послідовністю *S* розраховується статистика тесту *c(S)*;
- з використанням спеціальної функції та статистики тесту розраховується значення ймовірності $P = f\left(c\left(S\right)\right)$;
- значення ймовірності *P* порівнюється з пороговим значенням $\alpha \in [0,96; 0,99]$. Якщо $P \geq \alpha$, то гіпотеза H_0 приймається. В іншому випадку приймається альтернативна гіпотеза.

Пакет містить 15 статистичних тестів, але, фактично, в залежності від вхідних параметрів обчислюються 188 значень ймовірності *P*, які можна розглядати як результат роботи окремих тестів.

1) *Частотний побітовий тест*. Спрямовано на визначення співвідношення між нулями та одиницями у двійковій послідовності певної довжини. Для дійсно випадкової бінарної послідовності кількість нулів та одиниць майже однакова. Тест оцінює, наскільки близькою є доля одиниць до 0,5.
2) *Частотний блоковий тест*. Суть тесту полягає у визначенні долі одиниць всередині блоку довжиною *m* бітів, тобто необхідно з'ясувати, чи дійсно частота повторення одиниць в блоці довжиною *m* бітів приблизно є рівною $m/2$, як можна було б припустити у випадку випадкової послідовності.
3) *Тест на послідовність однакових бітів*. У цьому тесті відбувається пошук рядків, тобто неперервних послідовностей однакових бітів. Ряд (серія) довжиною *k* бітів складається з *k* абсолютно ідентичних бітів, починається та закінчується з біту, який містить протилежне значення. В даному тесті необхідно з'ясувати, чи дійсно кількість таких рядків відповідає їх кількості у випадковій послідовності. Зокрема, визначається швидко чи повільно чергуються одиниці та нулі у початковій послідовності.
4) *Тест на найдовшу послідовність одиниць в блоці*. В даному тесті визначається найдовший рядок одиниць всередині блоку довжиною m бітів. Необхідно з'ясувати, чи дійсно довжина такого рядка відповідає очікуванням довжини найдовшого рядку одиниць у випадку абсолютно випадкової послідовності.
5) *Тест рангів бінарних матриць*. Тут здійснюється розрахунок рангів неперетинних під матриць, побудованих з початкової двійкової послідовності. Метою цього тесту є перевірка на лінійну залежність підрядків фіксованої довжини, що складають початкову послідовність.
6) *Спектральний тест*. Суть тесту полягає в оцінці висоти піків дискретного перетворення Фур'є початкової послідовності. Метою є виявлення періодичних властивостей вхідної послідовності, наприклад, близько розташованих один до одного повторюваних ділянок. Ідея полягає в тому, щоб число піків, що перевищують порогове значення у 95 % за амплітудою, було значно більшим за 5 %.
7) *Тест на співпадіння шаблонів, що не перекриваються*. В даному тесті підраховується кількість заздалегідь визначених шаблонів, які знайдені в початковій послідовності. Необхідно виявити генератори випадкових або псевдовипадкових чисел, що формують занадто часто задані неперіодичні шаблони. Як і в тесті № 8 на співпадіння шаблонів, що перекриваються, для пошуку конкретних шаблонів довжиною *m* бітів

використовується вікно також довжиною m бітів. Якщо шаблон не знайдено, вікно зсувається на один біт. Якщо ж шаблон знайдено, тоді вікно пересувається на біт, який є наступним за знайденим шаблоном, та пошук продовжується далі.

8) *Тест на співпадіння шаблонів, що перекриваються*. Суть даного тесту полягає в підрахунку кількості заздалегідь визначених шаблонів, які знайдені в початковій послідовності. Як і в тесті № 7 на співпадіння шаблонів, що не перекриваються, для пошуку конкретних шаблонів довжиною m бітів використовується вікно також довжиною m бітів. Сам пошук проводиться аналогічним способом. Якщо шаблон не знайдено, вікно зсувається на один біт. Різниця між цим тестом та тестом № 7 полягає лише в тому, що коли шаблон знайдено, вікно пересувається тільки на один біт вперед, після чого пошук продовжується далі.
9) *Універсальний статистичний тест Маурера*. Тут визначається число бітів між однаковими шаблонами в початковій послідовності (міра, що має безпосереднє відношення до довжини стиснутої послідовності). Необхідно з'ясувати, чи може дана послідовність бути значно стиснута без втрат інформації. У разі, якщо це можливо зробити, то вона не є істинно випадковою.
10) *Тест на лінійну складність*. В основі тесту лежить принцип роботи лінійного регістра зсуву зі зворотним зв'язком. Необхідно з'ясувати, чи є вхідна послідовність досить складною для того, щоб вважатися абсолютно випадковою. Абсолютно випадкові послідовності характеризуються довгими лінійними регістрами зсуву зі зворотним зв'язком. Якщо ж такий регістр занадто короткий, то передбачається, що послідовність не є в повній мірі випадковою.
11) *Тест на періодичність*. Даний тест полягає в підрахунку частоти всіх можливих перекривань шаблонів довжини m бітів протягом початкової послідовності бітів. Метою є визначення, чи дійсно кількість появ $2m$ шаблонів, що перекриваються, довжиною m бітів, є приблизно таким як і у випадку абсолютно випадковою вхідної послідовності бітів. Остання, як відомо, володіє одноманітністю, тобто кожен шаблон довжиною m біт з'являється в послідовності з однаковою ймовірністю. Варто відзначити, що при $m = 1$ тест на періодичність переходить в частотний побітовий тест (№ 1).
12) *Тест приблизної ентропії.* Як і в тесті на періодичність, в даному тесті акцент робиться на підрахунку частоти всіх можливих перекривань шаблонів довжини m бітів протягом початкової послідовності бітів. Необхідно порівняти частоти перекривання двох послідовних блоків початкової послідовності з довжинами m та $m+1$ з частотами перекривання аналогічних блоків в абсолютно випадковій послідовності.
13) *Тест кумулятивних сум*. Тест полягає в максимальному відхиленні (від нуля) при довільному обході, визначеному кумулятивною сумою заданих $(-1,+1)$ цифр в послідовності. Необхідно визначити, чи є кумулятивна сума часткових послідовностей, що виникають у вхідний послідовності, занадто великою або занадто маленькою у порівнянні з очікуваною поведінкою такої суми для абсолютно випадкової вхідної послідовності. Таким чином, кумулятивна сума може розглядатися як довільний обхід. Для випадкової послідовності відхилення від довільного обходу повинні бути близько нуля.
14) *Тест на довільні відхилення*. Суть даного тесту полягає в підрахунку числа циклів, що мають суворо k відвідувань при довільному обході кумулятивної суми. Довільний обхід кумулятивної суми починається з часткових сум після послідовності $(0,1)$, перекладеної у відповідну послідовність $(-1,+1)$. Цикл довільного обходу складається з серії кроків одиничної довжини, виконаних у випадковому порядку. Крім того, такий обхід починається і закінчується на одному і тому ж елементі. Мета даного тесту полягає у визначенні того, чи відрізняється число відвідувань певного стану всередині циклу від аналогічного числа в разі абсолютно випадкової вхідної послідовності.

Фактично даний тест є набором, що складається з восьми тестів, які проводяться для кожного з восьми станів циклу: -4, -3, -2, -1 та +1, +2, +3, +4.

15) *Інший тест на довільні відхилення*. У цьому тесті підраховується загальна кількість відвідувань певного стану при довільному обході кумулятивної суми. Метою є визначення відхилень від очікуваного числа відвідувань різних станів при довільному обході. Насправді цей тест складається з 18 тестів, що проводяться для кожного стану: -9, -8, ..., -1 та +1, +2, ..., +9.

Таким чином, в результаті тестування двійкової послідовності формується вектор $P = \{P_1, P_2 ... P_{188}\}$ значень ймовірностей. Аналіз складових P_j цього вектору дозволяють вказати на конкретні дефекти випадковості протестованої послідовності.

Проходження кожного з 15 статистичних тестів є важливим критерієм оцінки псевдовипадкового генератору. Тому навіть не відповідність за одним чи більше критеріями означає, що ключовий потік не може на високому рівні протистояти криптоаналізу. Якщо, з іншого боку, генератор проходить всі тести, це зовсім не говорить про захищеність генератору, оскільки такі тести не враховують особливостей реальної конструкції генератору.

Накопичений досвід проведення статистичного тестування показує, що кількість пройдених тестів досліджуваним генератором безпосередньо залежить від вибраної вихідної послідовності криптоалгоритму. Для забезпечення заданої достовірності результатів статистичного тестування в роботі [20, 21] запропоновано оцінити математичне сподівання числа пройдених тестів X_i досліджуваним генератором (криптоалгоритмом), розглядаючи при цьому кожне i-е тестування як одне спостереження (досвід), тобто як конкретну реалізацію деякої випадкової величини X.

При проведенні статистичних досліджень за кожним алгоритмом було сформовано 100 послідовностей завдовжки 10^8 байтів, тобто розмір статистичної вибірки за кожним алгоритмом сягав 10^{10} байтів. Кожне тестування (за кожною із 100 послідовностей) розглядалося як незалежне спостереження. У табл. 5 наведено узагальнені результати статистичного тестування за кожним дослідженим алгоритмом.

Таблиця 5

Результати статистичного тестування алгоритмів ґешування

Назва алгоритму	M099	D099	S099	P099	M096	D096	S096	P096	MIN
ГОСТ 34.311	132.90	43.93	6.62	1	186.83	1.46	1.20	1	182
СТРИБОГ256	131.92	55.93	7.47	1	186.70	1.81	1.34	1	181
СТРИБОГ512	131.64	48.99	6.99	1	186.76	1.88	1.37	1	182
BALLOON 32	134.20	60.56	7.78	1	187.10	1.09	1.04	1	185
BALLOON 64	126.50	0.25	0.50	1	183.00	1.00	1.00	1	182
BLAKE256	133.31	55.13	7.42	1	186.75	1.90	1.38	1	183
BLAKE512	132.65	55.74	7.46	1	186.73	1.59	1.26	1	183
BMW	132.39	48.99	6.99	1	186.92	1.55	1.24	1	182
CUBEHASH	131.22	55.65	7.46	1	186.80	1.44	1.2	1	183
DJB-2	8.92	1.21	1.10	1	11.64	0.29	0.53	1	10
DJB-2 XOR	2.99	2.16	1.47	1	4.94	1.69	1.30	1	0
ECHO	131.96	53.85	7.33	1	186.51	2.16	1.47	1	182
FUGUE 224	133.07	45.78	6.76	1	186.61	2.11	1.45	1	180
FUGUE 256	132.42	43.74	6.61	1	186.78	2.25	1.50	1	180
FUGUE 384	131.03	57.22	7.56	1	186.66	1.78	1.33	1	182
FUGUE 512	133.02	62.31	7.89	1	186.76	2.14	1.46	1	180
GROESTL 256	133.23	56.01	7.48	1	186.72	2.14	1.46	1	181
GROESTL 512	133.01	58.58	7.65	1	187.14	0.90	0.94	1	184
HAMSI 224	132.84	53.65	7.32	1	186.66	2.36	1.53	1	112
HAMSI 256	131.71	51.42	7.17	1	186.87	1.87	1.36	1	181
HAMSI 384	132.86	51.26	7.15	1	187.19	1.21	1.10	1	182

Назва алгоритму	M099	D099	S099	P099	M096	D096	S096	P096	MIN
HAMSI 512	132.17	51.16	7.15	1	186.45	2.68	1.63	1	179
J-H	131.70	71.63	8.46	1	186.69	2.43	1.56	1	180
KECCAK 256	131.27	58.79	7.66	1	186.52	2.70	1.64	1	181
KECCAK 512	132.40	48.12	6.93	1	186.78	1.41	1.18	1	182
LOSELOSE	16.00	0.00	0.00	1	16.00	0.00	0.00	1	16
LUFFA	133.11	47.07	6.86	1	186.71	1.50	1.22	1	183
PROGPOW	130.00	0.00	0.00	1	188.00	0.00	0.00	1	188
RANDOMX	0.00	0.00	0.00	1	0.00	0.00	0.00	1	0
RIPEMD160	84.79	84.04	9.16	1	132.11	80.95	8.99	1	105
SCRYPT 1024	133.80	68.36	8.26	1	187.10	1.49	1.22	1	184
SCRYPT 16384	140.00	0.00	0.00	1	185.00	0.00	0.00	1	185
SHA2 256	132.70	57.39	7.57	1	186.74	1.67	1.29	1	182
SHA2 512	133.01	51.78	7.19	1	186.87	1.99	1.41	1	182
SHABAL 224	132.76	50.12	7.08	1	186.67	2.52	1.58	1	180
SHABAL 256	133.18	61.72	7.85	1	186.61	2.39	1.54	1	115
SHABAL 384	131.63	45.61	6.75	1	186.54	2.00	1.41	1	180
SHABAL 512	132.81	45.41	6.73	1	186.87	1.57	1.25	1	182
SHAVITE	132.01	53.72	7.33	1	186.9	1.53	1.23	1	182
SIMD	133.09	39.54	6.28	1	186.85	1.58	1.25	1	182
SKEIN	131.90	46.97	6.85	1	186.71	1.26	1.12	1	184
WHIRLPOOL	132.29	47.90	6.92	1	186.78	1.59	1.26	1	182
X11	132.46	46.48	6.81	1	186.80	1.28	1.13	1	184
X12	133.10	21.29	4.61	1	186.20	2.36	1.53	1	183
X13	137.00	74.56	8.63	1	186.90	0.69	0.83	1	185
X14	131.40	24.44	4.94	1	186.90	0.69	0.83	1	123
X15	130.10	31.49	5.61	1	186.20	4.56	2.13	1	182
X16	0.90	0.09	0.30	1	1.00	0.00	0.00	1	1
X17	3.20	0.36	0.60	1	5.80	0.16	0.40	1	5

В табл. 5 наведено такі дані:

– «M096» та «M099» – оцінки математичного сподівання (вибіркові середні) числа пройдених статистичних тестів за критерієм $P_j \geq 0,96$ та за критерієм $P_j \geq 0,99$, відповідно;

– «D096» та «D099» («S096» та «S099») – оцінки дисперсій (середньоквадратичних відхилень) результатів тестування числа пройдених статистичних тестів за критеріями $P_j \geq 0,96$ та $P_j \geq 0,99$, відповідно;

– «P099» – значення довірчої ймовірності для числа пройдених статистичних тестів за критерієм $P_j \geq 0,99$ та при точності $\varepsilon = 2$;

– «P096» – значення довірчої ймовірності для числа пройдених статистичних тестів за критерієм $P_j \geq 0,96$ та при точності $\varepsilon = 1$;

– «Min096» мінімальні значення числа пройдених статистичних тестів за критерієм $P_j \geq 0,96$.

Результати статистичних досліджень (статистичні портрети) алгоритмів ґешування наведено на рис. 3 – 42, де по шкалі абсцис відмічений номер статистичного тесту, а за шкалою ординат – ймовірність проходження тесту.

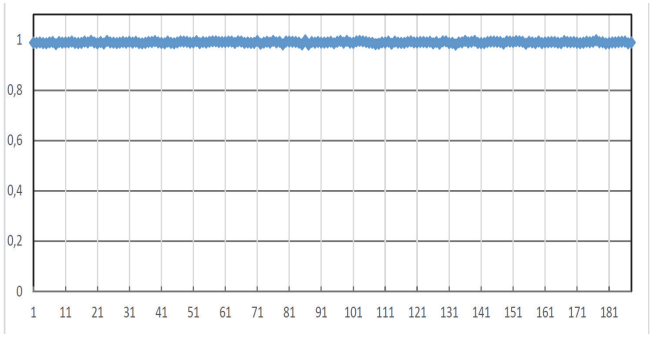

Рис. 3. BALLOON32

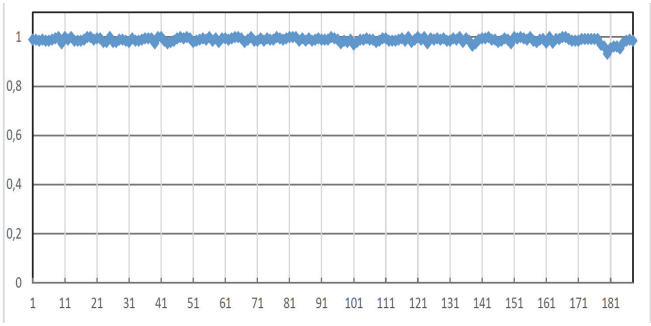

Рис. 4. BALLOON64

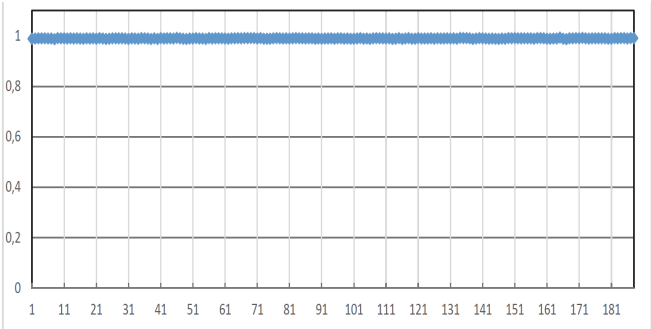

Рис. 5. BLAKE 256

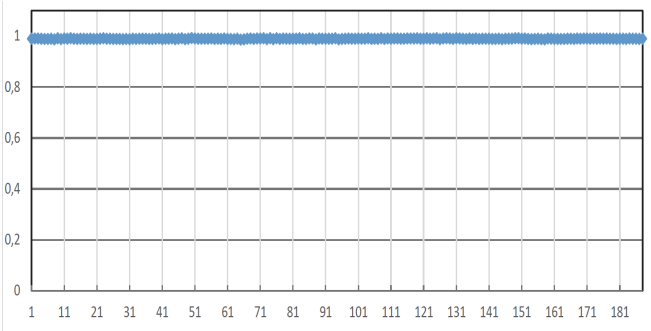

Рисунок 6. BLAKE 512

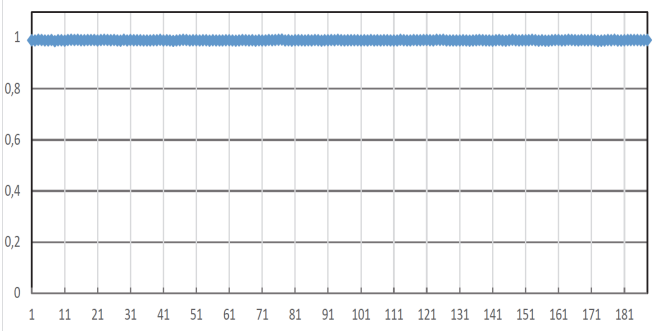

Рис. 7. BMW

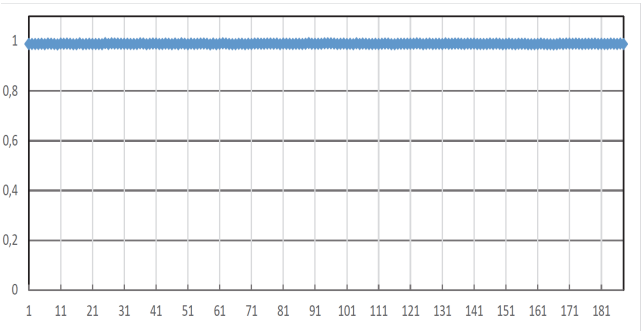

Рис. 8. CUBEHASH

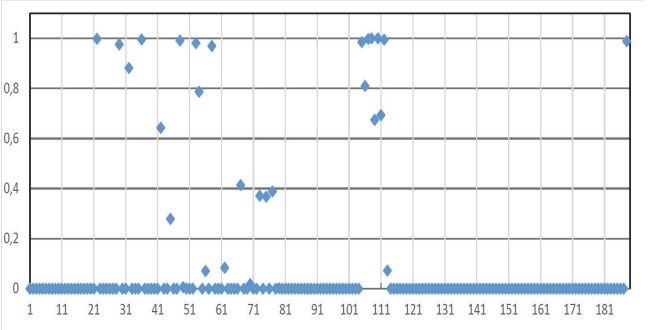

Рис. 9. DJB-2

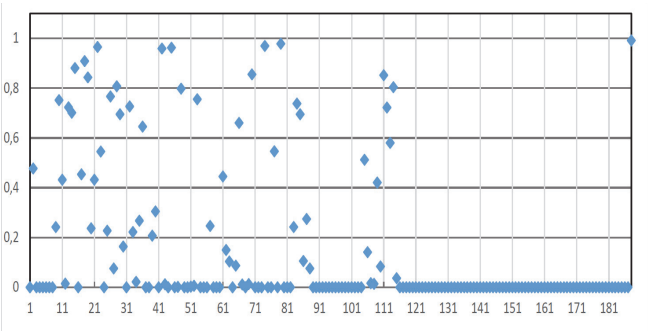

Рис. 10. DJB-2XOR

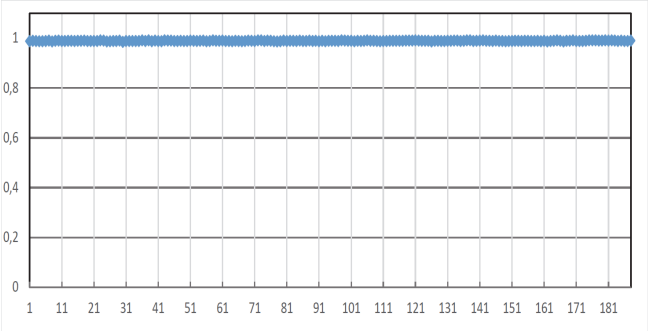

Рис. 11. ECHO

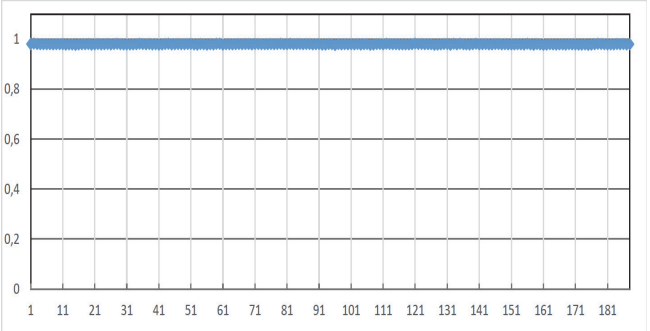

Рис. 12. KECCAK 256

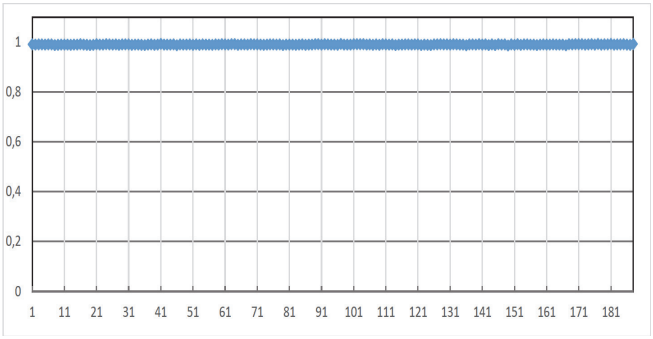

Рис. 13. KECCAK 512

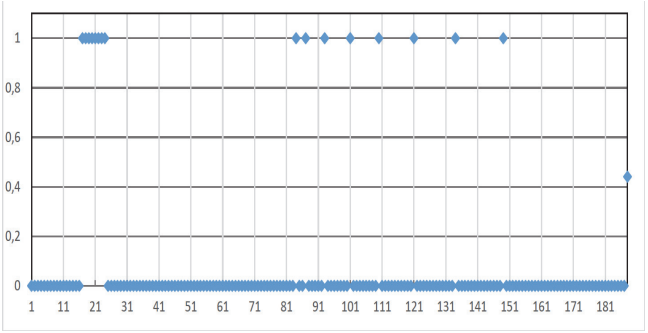

Рис. 14. LOSELOSE

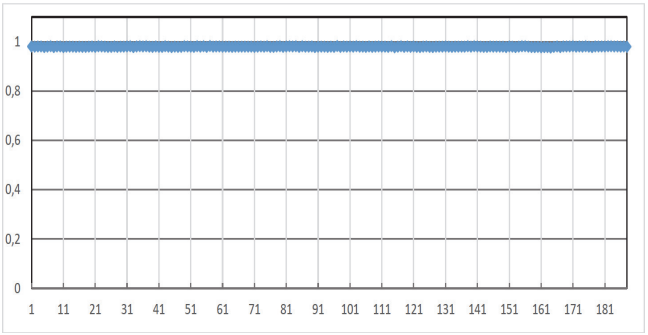

Рис. 15. LUFFA

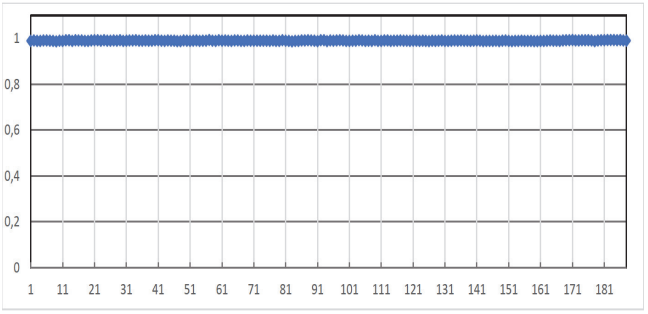

Рис. 16. FUGUE224

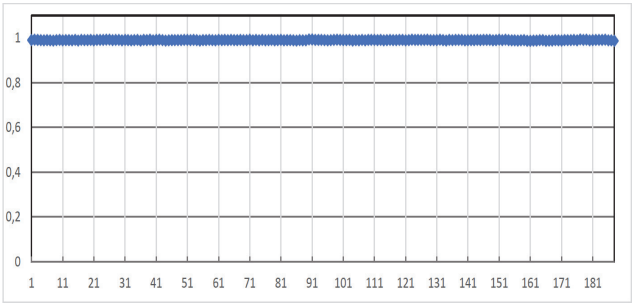

Рис. 17. FUGUE256

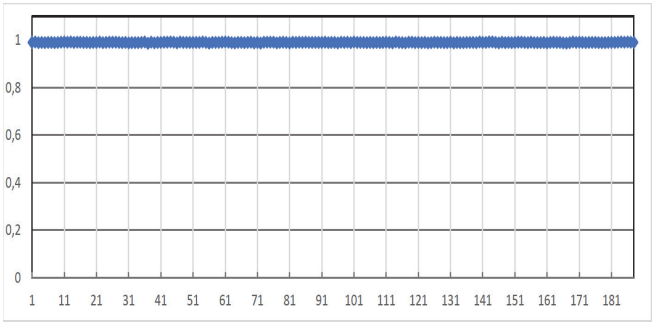

Рис. 18. FUGUE384

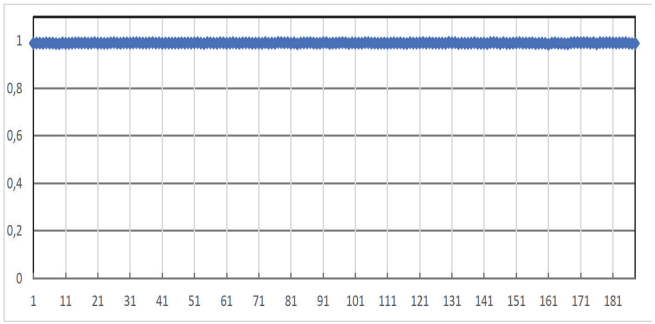

Рис. 19. FUGUE512

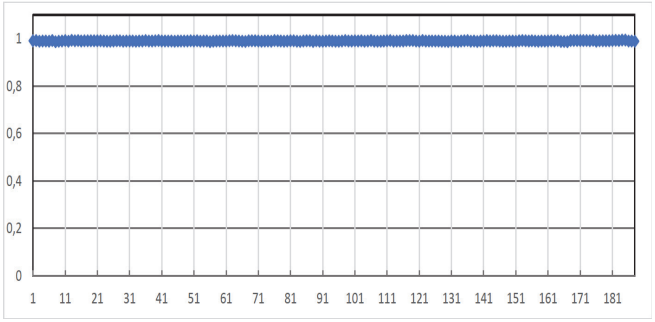

Рис. 20. GOST_256

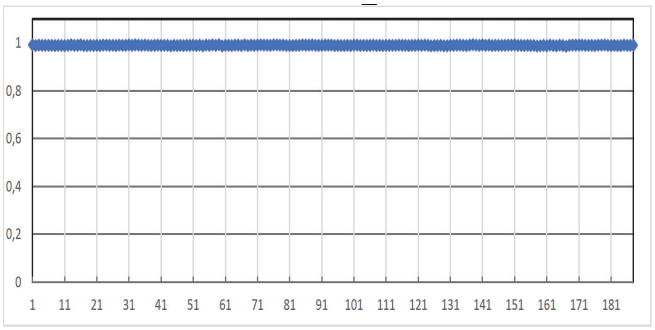

Рис. 21. Stribog_512

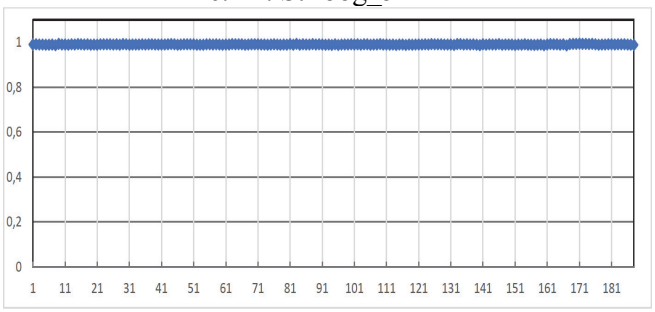

Рис. 22. WHIRLPOOL512

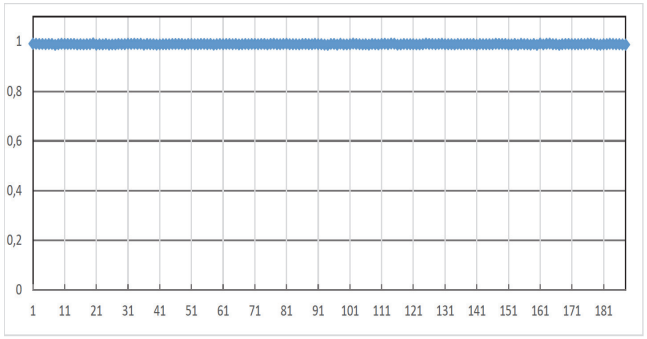

Рис. 23. GROESTL 256

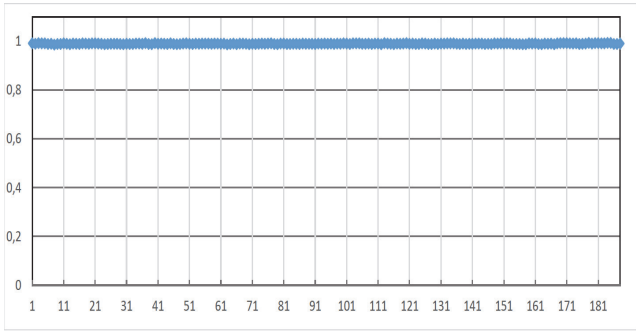

Рис. 24. GROESTL 512

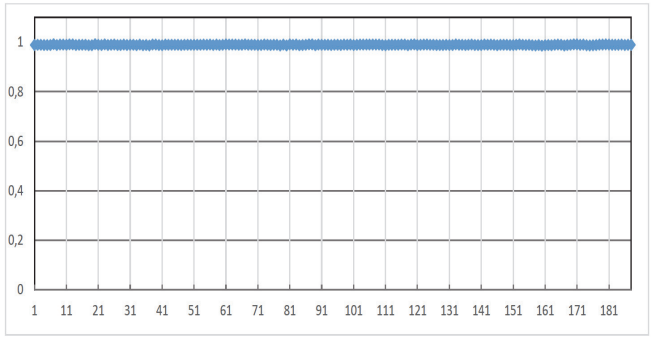

Рис. 25. HAMSI 224

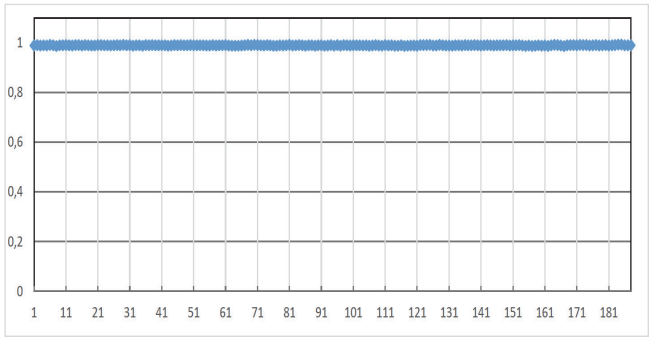

Рис. 26. – HAMSI 256

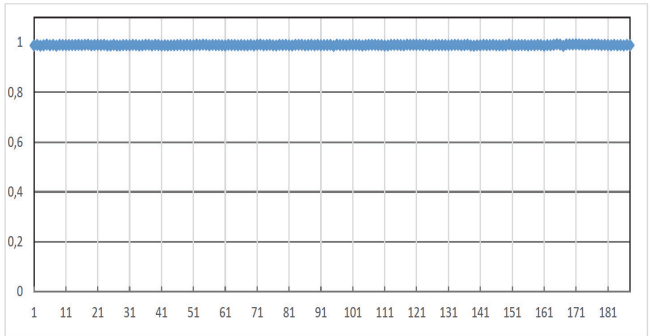

Рис. 27. HAMSI 384

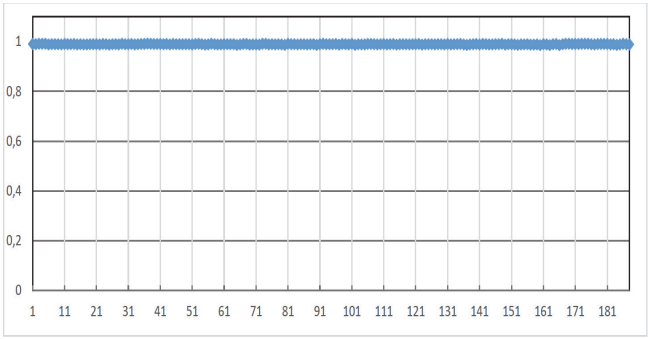

Рис. 28. HAMSI 512

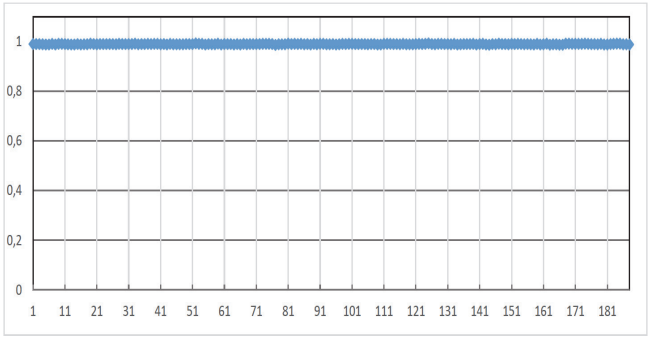

Рис. 29. – J-H

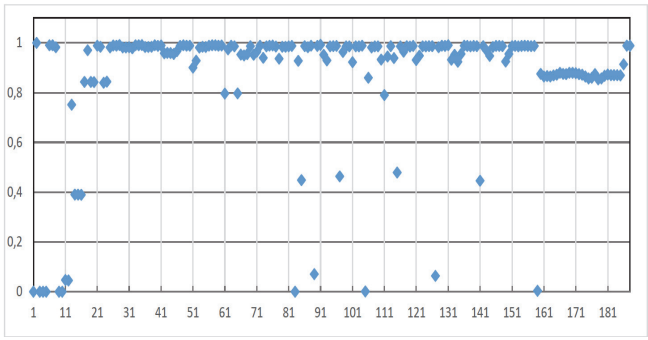

Рис. 30. RIPEMD160

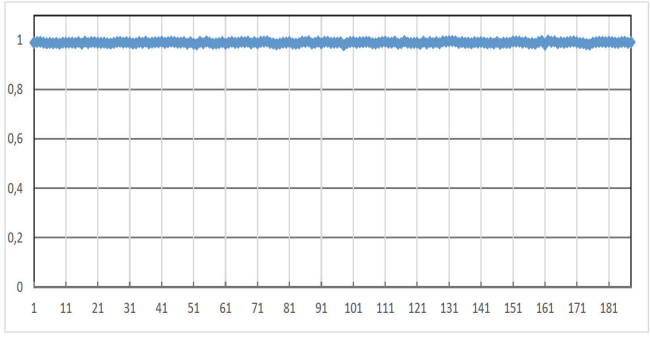

Рис. 31 – SCRYPT 1024

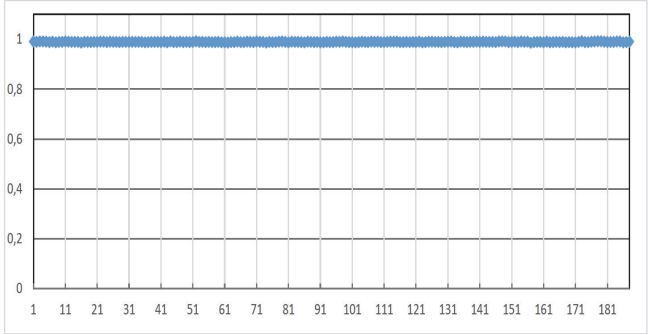

Рис. 32. SHA2 256

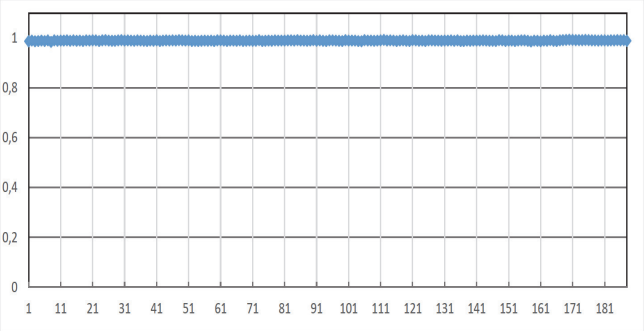

Рис. 33. – SHA2 512

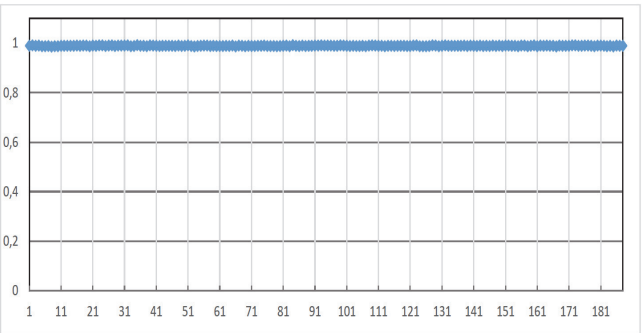

Рис. 34. SHABAL 224

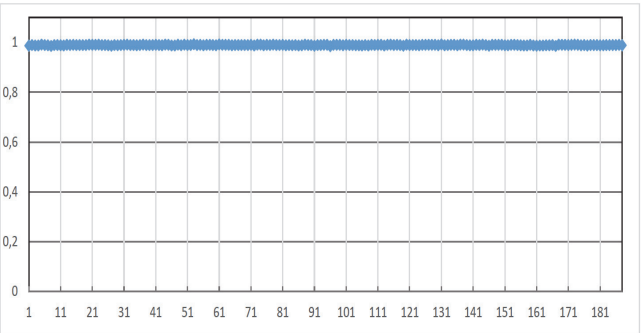

Рис. 35. SHABAL 256

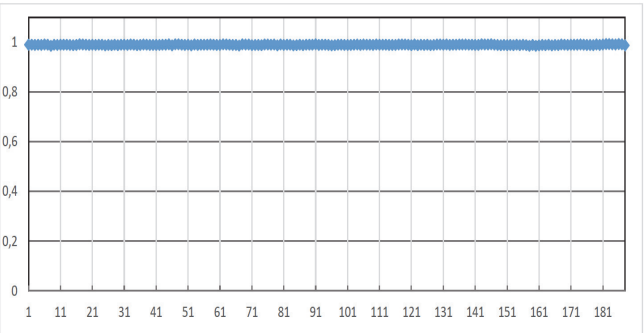

Рис. 36. SHABAL 384

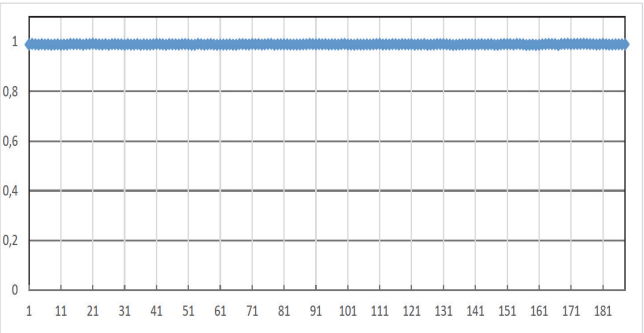

Рис. 37. SHABAL 512

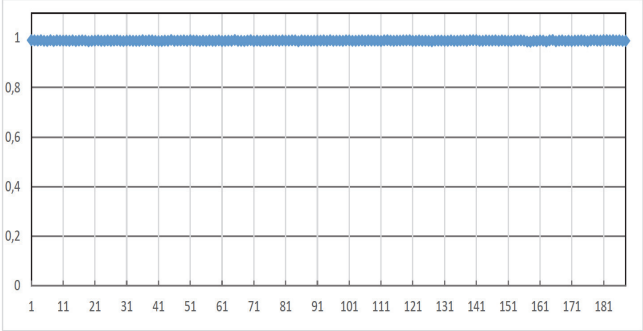

Рис. 38. SHAVITE

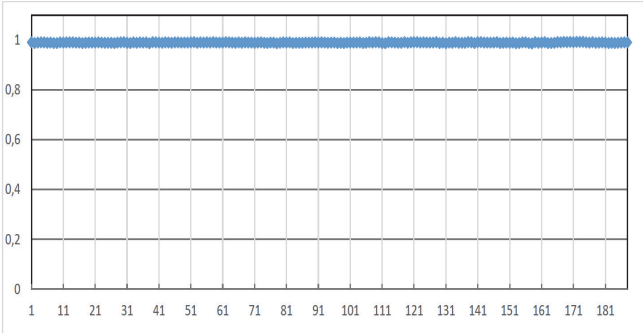

Рис. 39. SIMD

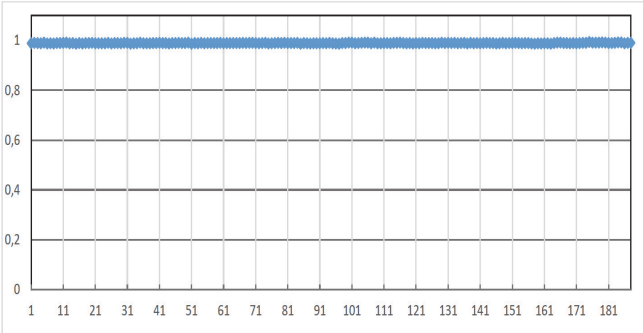

Рис. 40. SKEIN

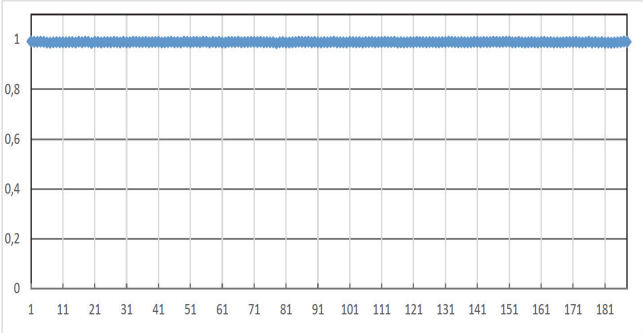

Рис. 41. STREEBOG 256

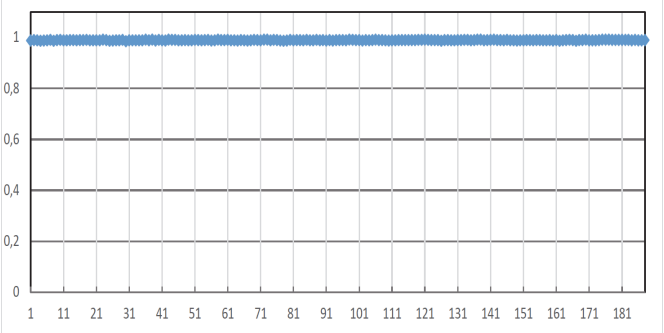

Рис. 42. X11

Отримані результати статистичних досліджень свідчать, що певні швидкісні алгоритми не можуть бути застосовані у криптографічних додатках. Це стосується, наприклад, алгоритмів DJB-2, LOSELOSE та інших, бо ці алгоритми, фактично, обчислюють не криптографічну контрольну суму. Але більшість з досліджених алгоритмів криптографічного ґешування показали високі статистичні властивості і за критерієм нерозрізнюваності із випадковою послідовністю мають високі показники.

Висновки

Функції ґешування являють собою складний і дуже важливий криптографічний примітив, який застосовується практично в усіх механізмах та протоколах криптографічного захисту інформації (формування паролів, шифрування, генерація псевдовипадкових послідовностей, формування електронного підпису, тощо). Останніми роками коло застосування ґешування значно розширилося. Зокрема, із появою та стрімким розповсюдженням децентралізованих розподілених систем, побудованих за технологією так званих «зв'язаних списків» (блокчейн-системи) виникла гостра потреба у швидких, безпечних та надійних функціях ґешування, бо саме на їх властивостях непередбачуваності та необоротності будуються захищені блокчейн-ланцюжки. Задача обрання ґеш-функції значно ускладняється через поширення спеціалізованих обчислювачів, які розробляються та практично застосовуються для пошуку прообразів наперед сформованих ґеш-значень (ASIC-майнінг). Інвестуючи в придбання ASIC-обчислювачів окремі гравці можуть бути поставлені у свідомо більш вигідне становище порівняно з іншими користувачами блокчейн-системи і, отже, можуть стати причиною недовіри та компрометації децентралізованих технологій (наприклад, різних криптовалют, розподілених сховищ, смарт-контрактів, тощо). Отже дослідження властивостей сучасних алгоритмів ґешування та обґрунтування рекомендацій щодо їх застосування для розбудови національного сегменту блокчейн-технологій є безумовно важливим та надзвичайно актуальним науковим завданням.

В цій завершальній статті (із серії робіт, присвячених алгоритмам ґешування у сучасних блокчейн-системах) проведено порівняльні дослідження як стандартизованих на міжнародному та національному рівні алгоритмів ґешування, так і ґеш-функцій, що були представлені на різних криптографічних конкурсах на науково-пошукових проектах. Зокрема в табл. 1 – 5 та на рис. 1 – 42 наведено отримані результати з дослідження таких алгоритмів: ГОСТ 34.311, СТРИБОГ256, СТРИБОГ512, BALLOON 32, BALLOON 64, BLAKE256, BLAKE512, BMW, CUBEHASH, DJB-2, DJB-2 XOR, ECHO, FUGUE 224, FUGUE 256, FUGUE 384, FUGUE 512, GROESTL 256, GROESTL 512, HAMSI 224, HAMSI 256, HAMSI 384, HAMSI 512, J-H, KECCAK 256, KECCAK 512, LOSELOSE, LUFFA, PROGPOW, RANDOMX, RIPEMD160, SCRYPT 1024, SCRYPT 16384, SHA2 256, SHA2 512, SHABAL 224, SHABAL 256, SHABAL 384, SHABAL 512, SHAVITE, SIMD, SKEIN, WHIRLPOOL, X11.

Для всіх розглянутих алгоритмів були проведені порівняльні дослідження швидкодії на різних обчислювальних платформах та із різними вхідними параметрами. Отримані результати частково наведено у табл. 1 – 4. Зокрема встановлено, що більшість проектів розподілених мереж використовує надійні та перевірені часом алгоритми криптографічного ґешування (наприклад, алгоритми KECCAK, SHA2, RIPEMD160, тощо), які володіють відносно високими швидкісними показниками. Але останніми роками для захисту від ASIC-майнерів почали застосовуватися і інші ґеш-функції, які хоча і можуть бути навіть швидшими за KECCAK, SHA2 або RIPEMD160, але володіють певними вразливостями стосовно властивостей необоротності. Як приклад можна навести застосування алгоритмів MD4, EDONR-256, EDONR-512, ED2K, або навіть найпростіших функцій DJB-2 та LOSELOSE. Через простоту обчислення певних показників не можна нехтувати порушенням властивостей необоротності, особливо якщо саме на них базуються основні переваги блокчейн-мереж. Отримані результати порівняльного аналізу дають можливість обирати криптографічні функції ґешування за критеріями швидкодії на різних пристроях та обгартовувати їх практичне застосу-

вання для побудови децентралізованих систем типу блокчейн. Це також стосується і результатів порівняльного аналізу швидкодії на графічних обчислювачів, особливо з приводу можливої розбудови національного сегменту блокчейн-мереж.

Отримані результати статистичної безпеки (див. табл. 5 та рис. 3 – 42) свідчать, що більшість функцій ґешування задовольняють встановленим критеріям (за методикою NIST STS), тобто за різними показниками вихідні послідовності (ґеш-значення) не відрізняються (у статистичному сенсі) від реалізації випадкового процесу. Це стосуються, переважно, відомих та стандартизованих алгоритмів, які застосуються в різних криптографічних додатках та вже були суттєво досліджені та вивчені при попередніх випробуваннях. Але серед алгоритмів із табл. 5 є і такі, показники статистичної безпеки яких є незадовільними, або зовсім неприйнятними. Наприклад, відомий алгоритм ґешування RIPEMD160, який стандартизовано в ISO/IEC 10118-3:2018 та прийнято до застосування в Європейському Союзі, показав невисокі значення статистичної безпеки (середнє число пройдених статистичних тестів за критерієм $P_j \geq 0,96$ не перевищує 85). Тобто, якщо на вхід алгоритму RIPEMD160 подається надмірна послідовність (в наших дослідженнях вхідна послідовність формувалася звичайним лічильником), формовані ґеш-коди за окремими тестами відрізняються від випадкової послідовності, тобто мають певний детермінізм. І хоча нами не знайдено конкретних дефектів алгоритму RIPEMD160, отримані результати свідчать про певні вади формованих ґеш-кодів з точки зору їх випадковості та непередбачуваності. Окремо слід відмітити незадовільні показники статистичної безпеки алгоритмів ґешування DJB-2, DJB-2 XOR та LOSELOSE. Ці алгоритми показали найвищі показники швидкодії (див. табл. 1, 2), але за показниками статистичної безпеки вони є неприйнятними до практичного застосування у криптографічних додатках. Цей висновок є передбачуваним, бо алгоритми DJB-2, DJB-2 XOR та LOSELOSE по суті не є криптографічними, обчислення ґеш-кодів в них є подібним до звичайної контрольної суми. Але, як показують отримані результати, навіть у разі використання статистично-небезпечних алгоритмів у складі каскадних схем майнінгу (наприклад, у складі алгоритмів ґешування сімейства «X»), формовані ґеш-значення також не задовольняють показникам статистичної безпеки (див. останні дві строки таблиці 5).

Таким чином, вибір алгоритму ґешування для побудови елементів блокчейн-систем є надзвичайно важливим і кропітким. З огляду на отримані результати окрім показників швидкодії необхідно також враховувати надійність та безпеку криптоперетворень. Важливим є також наявність спеціалізованих обчислювачів (ASIC), застосування яких значно прискорює майнінг у певних протоколах консенсусу. Отже для обґрунтування вибору алгоритмів ґешування необхідно враховувати різні фактори та показники ефективності, в тому числі особливості побудови конкретної блокчейн-системи, протоколів консенсусу, алгоритмів обробки та обміну повідомленнями, тощо.

Список літератури:

1. Bernstein hash djb2. Електронний ресурс. Режим доступу: https://riot-os.org/api/group__sys__hashes__djb2.html
2. The C Programming Language by Brian W. Kernighan (1978-02-22) Paperback, Prentice Hall, 178 p.
3. Hash Functions. Created January 04, 2017, Updated May 03, 2019. Електронний ресурс. Режим доступу: https://csrc.nist.gov/projects/hash-functions/sha-3-project
4. Classification of the SHA-3 Candidates. By Ewan Fleischmann, Christian Forler, and Michael Gorski. Version 0.90, April 19, 2009. Електронний ресурс. Режим доступу: https://eprint.iacr.org/2008/511.pdf
5. Ed2k-hash. 7 May 2005. Електронний ресурс. Режим доступу: https://wiki.anidb.info/w/Ed2k-hash
6. ed2k-tools. Tools for eDonkey2000 and Overnet. Електронний ресурс. Режим доступу:http://ed2k-tools.sourceforge.net/index.shtml
7. Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD. By Xiaoyun Wang , Dengguo Feng, Xuejia Lai3, Hongbo Yu. August 17, 2004. Електронний ресурс. Режим доступу: http://eprint.iacr.org/2004/199.pdf
8. The hash function RIPEMD-160. Електронний ресурс. Режим доступу: http://homes.esat.kuleuven.be/~bosselae/ripemd160.html

9. Secure Hash Standard. Federal Information. Processing Standards Publication 180-2. 2002 August 1. (FIPS PUB 180-2) Електронний ресурс. Режим доступу: http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf

10. Keccak hashing algorithm (SHA-3) – Keccak Coins and miner for Keccak. Електронний ресурс. Режим доступу: https://coinguides.org/keccak-algorithm-miner-coins/

11. NIST Selects Winner of Secure Hash Algorithm (SHA-3) Competition. Created October 02, 2012, Updated December 11, 2018. Електронний ресурс. Режим доступу: https://www.nist.gov/news-events/news/2012/10/nist-selects-winner-secure-hash-algorithm-sha-3-competition

12. ГОСТ Р 34.11-2012. Информационная технология. Криптографическая защита информации. Функция хэширования. Дата введения 2013-01-01. Електронний ресурс. Режим доступу: http://docs.cntd.ru/document/gost-r-34-11-2012

13. A New Standard of Ukraine: The Kupyna Hash Function. Roman Oliynykov1, Ivan Gorbenko, Oleksandr Kazymyrov,Victor Ruzhentsev, Oleksandr Kuznetsov, Yurii Gorbenko, Artem Boiko, Oleksandr Dyrda, Viktor Dolgov, Andrii Pushkaryov. Електронний ресурс. Режим доступу: https://eprint.iacr.org/2015/885.pdf

14. Argon2. By Dmitry Khovratovich. 30 March 2015. Електронний ресурс. Режим доступу: https://www.cryptolux.org/index.php/Argon2

15. Алгоритм X13 для майнинга на графических процесорах / Александр Марков. 28 мая 2018. Електронний ресурс. Режим доступу: https://miningbitcoinguide.com/mining/sposoby/x13

16. Colin Percival. Stronger key derivation via sequential memory-hard functions. 2009. Електронний ресурс. Режим доступу: https://en.bitcoinwiki.org/wiki/Scrypt http://www.tarsnap.com/scrypt/scrypt.pdf

17. Hashcat. Advanced Password Recovery. Електронний ресурс. Режим доступу: http://hashcat.net/hashcat/

18. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. – Електронний ресурс. Режим доступу: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf

19. NIST Cryptographic Toolkit. Електронний ресурс. Режим доступу: https://csrc.nist.gov/projects/random-bit-generation/documentation-and-software

20. Кузнецов А.А., Мордвинов Р.И., Колованова Е.П., Самойлова А.В. Методика статистического тестирования криптографических алгоритмов // Спеціальні телекомунікаційні системи та захист інформації. Київ, 2014. №1(25). С.54-61

21. Кузнецов О.О., Луценко М.С., Андрушкевич А.В., Мелкозерова О.М., Новикова Д.В., Лобан А.В. Статистичні дослідження сучасних потокових шифрів // Прикладная радиоэлектроника. Харьков : ХНУРЭ, 2016. Т. 15. №3. С. 167 – 178.

Харківський національний
університет імені В.Н. Каразіна;
АТ «Інститут інформаційних технологій», м.Харків

Надійшла до редколегії 05.09.2019