

KPQClean 프로젝트

정보컴퓨터공학과 권혁동

KPQC란?

- Korea Post-Quantum Cryptography 프로젝트
- **한국형 양자내성암호 표준**을 위해 발족
 - NIST PQC와 동일한 목적이지만, **kpqC는 국내 표준**을 위한 프로젝트
- 22년 10월에 개최되어, 1라운드가 진행 중에 있음
 - PKE에 7개 알고리즘, DSA에 9개 알고리즘이 선정됨
- 23년 11월에 2라운드 진출 알고리즘이 발표 될 예정

KPQClean

- KPQClean은 현재 진행중인 kpqc를 대상으로 하는 프로젝트
 - 외부 과제와 약간의 연결고리 존재
- PQClean과 비슷한 목표를 대상으로 함
 - 모든 PQC 후보군에 대한 **의존성 제거**
 - 동일한 환경에서 **벤치마킹**
- PQC 공모전에 대한 관심을 이끌 수 있음

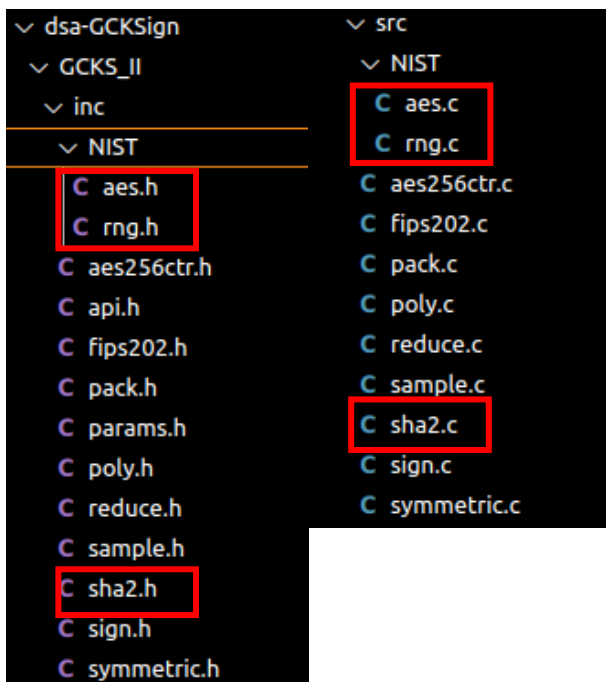
KPQClean

- 동일한 테스트 환경을 갖추기 위해 1개의 기기에서 실험
 - OS, CPU, RAM, GPU 모든 자원이 동일
 - IDE는 VS Code를 사용, 하지만 편집 기능만 사용하고 컴파일은 무관
 - 컴파일은 gcc로 통일 (11.3.0)
- 추후 변경될 수 있지만, 그래도 모든 알고리즘이 동일
- 최적화 레벨: -O2

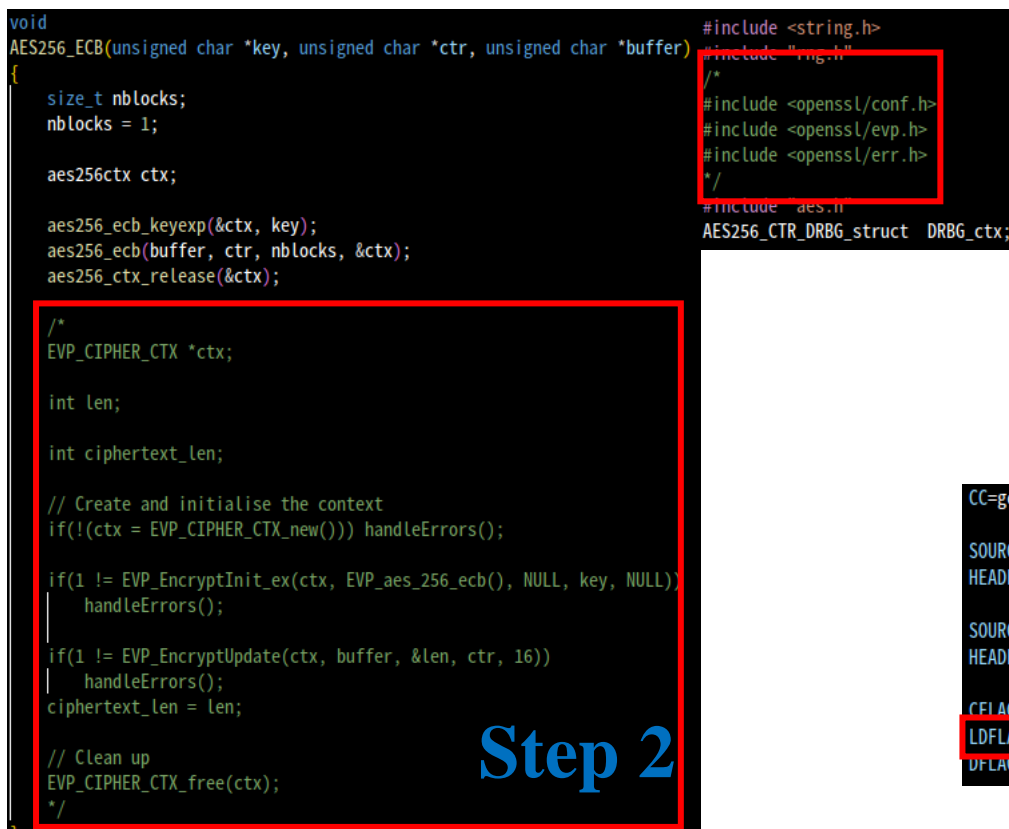
```
Makefile - kpqclean-Round1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
C main.c Makefile dsa-HAETAE Makefile _GCKS_11_
ds-GCKSign > GCKS_11 > Makefile
1 INCDIR := inc
2 SRCDIR := src
3
4 CC=gcc
5
6 SOURCES= ./src/fips202.c ./src/NIST/rng.c ./src/pack.c ./src/symmetric.c ./src/aes256ctr.c ./src/poly.c ./src/reduce.c ./src/sample.c ./src/sign.c ./src/NIST/aes.c ./src/sha2.c
7 HEADERS= ./inc/api.h ./inc/fips202.h ./inc/NIST/rng.h ./inc/pack.h ./inc/symmetric.h ./inc/aes256ctr.h ./inc/poly.h ./inc/reduce.h ./inc/sample.h ./inc/sign.h ./inc/params.h ./inc/NIST/aes.h
8
9 SOURCE = main.c ./src/*.* ./$(SRCDIR)/NIST/*.*
10 HEADER = ./$(INCDIR) ./$(INCDIR)/NIST/*.*
11
12 CFLAGS = -march=native -mtune=native -O3 -fomit-frame-pointer
13 LDFLAGS =
14 DFLAGS = -g3 -DDEBUG
15
16
17
18
19 all:
20 $(CC) $(CFLAGS) -c $(SOURCES) $(HEADERS)
21 $(CC) $(CFLAGS) *.o -o main $(LDFLAGS)
22 #rm *.o
23
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
gcc -Wall -Wextra -march=native -O3 -c -o poly.o poly.c
gcc -Wall -Wextra -march=native -O3 -c -o precomp.o precomp.c
gcc -Wall -Wextra -march=native -O3 -c -o randombytes.o randombytes.c
gcc -Wall -Wextra -march=native -O3 -c -o rng.o rng.c
gcc -Wall -Wextra -march=native -O3 -c -o samplerZ.o samplerZ.c
gcc -Wall -Wextra -march=native -O3 -c -o shake.o shake.c
gcc -Wall -Wextra -march=native -O3 -c -o sign.o sign.c
gcc -Wall -Wextra -march=native -O3 -c -o test_dist.o test_dist.c
gcc -Wall -Wextra -march=native -O3 -c -o vrfy.o vrfy.c
gcc main.o PQ_bench.o benchmarks.o codec.o common.o cpucycles.o falcon_keygen.o fft.o fips202.o fpr.o keygen.o nist.o normaldist.o poly.o precomp.o randombytes.o rng.o samplerZ.o shake.o sign.o test_dist.o vrfy.o
KAT/generator/PQGenKAT_sign.o KAT/generator/katrng.o -ln -o main
/usr/bin/ld: cpucycles.o: in function 'cpucycles':
cpucycles.c:(.text+0x0): multiple definition of 'cpucycles'; PQ_bench.o:PQ_bench.c:(.text+0x0): first defined here
/usr/bin/ld: KAT/generator/PQGenKAT_sign.o: in function 'main':
PQGenKAT_sign.c:(.text.startup+0x0): multiple definition of 'main'; PQ_bench.o:PQ_bench.c:(.text.startup+0x0): first defined here
collect2: error: ld returned 1 exit status
make: *** [: main] 오류 1
ld64 -T674768 -/kpqclean-Round1/dsa-SOLMAE/SoLMAE512_variable_key
History restored
ld64 -T674768 -/kpqclean-Round1/dsa-SOLMAE/SoLMAE512_variable_key
```

KPQClean

- 다양한 의존성이 존재, 그 중에서 라이브러리 의존성을 우선적으로 제거
 1. 외부 라이브러리는 구현은 편리하지만, 실행은 다소 복잡
 2. 사용한 라이브러리에 따라서 성능이 달라질 수 있음



Step 1



Step 2



Step 3

KPQClean

- 성능 측정용 코드를 작성
- Makefile에 성능 측정용 규칙 생성
 - 해당 규칙으로 컴파일 후 실행
 - 가동에 소요된 **평균 cycle**을 사용
- 대부분 PQC 후보는 **중앙값** 사용

```
#define TEST_LOOP 10000

uint64_t t[TEST_LOOP];

int64_t cpucycles(void)
{
    unsigned int hi, lo;

    __asm__ __volatile__ ("rdtsc\n\t" : "=a" (lo), "=d" (hi));

    return ((int64_t)lo) | (((int64_t)hi) << 32);
}

int PQC_bench(void)
{
    int r;
    double rejw=.0, rejyz=.0, rejctr=.0, rejctrkg=.0;
    unsigned long long i, j;

    unsigned char pk[CRYPTO_PUBLICKEYBYTES];
    unsigned char sk[CRYPTO_SECRETKEYBYTES];

    unsigned char m[100] = "kpqc benchm";
    unsigned char sm[CRYPTO_BYTES + 200];
```

```
PQCgenKAT_sign: ./PQCgenKAT_sign.c
$(CC) $(CFLAGS) -o $@ ./PQCgenKAT_sign.c $(HEADER) $(SOURCES) $(LDFLAGS)
./PQCgenKAT_sign
```

```
PQC_bench: ./PQC_bench.c
$(CC) -march=native -mtune=native -O2 -fomit-frame-pointer -fstack-usage -o $@ ./PQC_bench.c $(HEADER) $(SOURCES) $(LDFLAGS)
./PQC_bench
```

```
Module_bench: ./Module_bench.c
$(CC) -march=native -mtune=native -O2 -fomit-frame-pointer -o $@ ./Module_bench.c $(HEADER) $(SOURCES) $(LDFLAGS)
./Module_bench
```

```
BENCHMARK ENVIRONMENTS =====
CRYPTO_PUBLICKEYBYTES: 1760
CRYPTO_SECRETKEYBYTES: 288
CRYPTO_BYTES: 1952
Number of loop: 10000
KeyGen //////////////////////////////////////
KeyGen runs in ..... 191048 cycles
Sign //////////////////////////////////////
Sign runs in ..... 812492 cycles
Verify //////////////////////////////////////
Verify runs in ..... 175823 cycles
=====
hd@hd-TF67476HS:~/kpqclean-Round1/dsa-GCKSign/GCKS_II$
```

KPQClean

- 평균값(average)과 중앙값(median)의 차이?
 - 평균은 산술 평균으로 주어진 값 범위에서 일반적인 값을 표현
 - 중앙값은 위치 평균으로 값을 동일한 두 부분으로 나누는 기준
 - 평균은 데이터 세트가 정규분포를 따를 때 사용하기 적합
 - 중앙값은 데이터 세트에 왜곡이 심할 때 사용하기 적합
- PQC 후보들은 대부분 중앙값을 사용
 - 어느 것이 더 명확한 표현인지 검증은 필요함

평균 856.6	200	중앙 350
	250	
	280	
	330	
	350	
	600	
	1000	
	1700	
	3000	

KPQClean

- 성능 측정의 이상한 점
 - 일부 알고리즘은 성능이 확 튀는 경우가 존재

알고리즘	보안레벨	키 설정(cycles)	암호화(cycles)	복호화(cycles)
IPCC	1	15,006,320	236,123,169	2,859,824
	3	15,411,372	942,886	2,829,830
	4	15,265,594	1,137,671	3,248,358

알고리즘	보안레벨	키 설정(cycles)	서명생성(cycles)	검증(cycles)
pqsigRM	612	6,175,797,126	14,737,341	2,441,347
	613	58,885,289,264	2,145,199	1,061,605

- 백서에 기재된 파라미터와 소스코드에 작성된 파라미터가 다름
 - 개발자들과 논의가 필요해 보임

알고리즘	공개키(bytes)	암호문(bytes)	문서상 공개키	문서상 암호문
IPCC	3,600	322,000	4,800	92,000
	3,600	322,000	4,800	92,000
	3,600	322,000	4,800	92,000

알고리즘	공개키(bytes)	비밀키(bytes)	문서상 공개키	문서상 비밀키
AlMer	33	49	32	16
	49	73	48	24
	65	97	64	32

KPQClean

- KPQC github에 소스코드와 측정 결과를 게시하는 중
 - 6월 14일 KPQC built-in groups에 공개하는 것을 목표

<div>WG0</div> <div>Working Group 0 Commit</div> <div>LICENSE</div> <div>Initial commit</div> <div>README.md</div> <div>Update README.md</div> <div>README.md</div> <div><h2>KPQClean</h2><p>Benchmark on Korean Post Quantum Cryptography!</p><p>PQClean...</p><p>Performance in Table...</p><p>KEM Cycle PK SK Ciphertext</p><table><tr><th>First Header</th><th>Second Header</th></tr><tr><td>Content Cell</td><td>Content Cell</td></tr><tr><td>Content Cell</td><td>Content Cell</td></tr></table><p>Sign Cycle PK SK Signature</p><p>WG0 algorithm + descption... WG1 WG2</p><p>How to use</p><p>How to contribute Hyeok email... Hwajeong email...</p></div>	First Header	Second Header	Content Cell	Content Cell	Content Cell	Content Cell	<table><tr><th>Name</th><th>Last commit message</th></tr><tr><td>..</td><td></td></tr><tr><td>dsa-AIMer</td><td>Working Group 0 Commit</td></tr><tr><td>dsa-GCKSign</td><td>Working Group 0 Commit</td></tr><tr><td>dsa-HAETAE_revised</td><td>Working Group 0 Commit</td></tr><tr><td>dsa-MQSign</td><td>Working Group 0 Commit</td></tr><tr><td>dsa-NCCSign</td><td>Working Group 0 Commit</td></tr><tr><td>dsa-Peregrine</td><td>Working Group 0 Commit</td></tr><tr><td>pke-IPCC</td><td>Working Group 0 Commit</td></tr><tr><td>pke-NTRUplus</td><td>Working Group 0 Commit</td></tr><tr><td>pke-SMAUG_revised</td><td>Working Group 0 Commit</td></tr><tr><td>pke-TiGER</td><td>Working Group 0 Commit</td></tr></table>	Name	Last commit message	..		dsa-AIMer	Working Group 0 Commit	dsa-GCKSign	Working Group 0 Commit	dsa-HAETAE_revised	Working Group 0 Commit	dsa-MQSign	Working Group 0 Commit	dsa-NCCSign	Working Group 0 Commit	dsa-Peregrine	Working Group 0 Commit	pke-IPCC	Working Group 0 Commit	pke-NTRUplus	Working Group 0 Commit	pke-SMAUG_revised	Working Group 0 Commit	pke-TiGER	Working Group 0 Commit
First Header	Second Header																														
Content Cell	Content Cell																														
Content Cell	Content Cell																														
Name	Last commit message																														
..																															
dsa-AIMer	Working Group 0 Commit																														
dsa-GCKSign	Working Group 0 Commit																														
dsa-HAETAE_revised	Working Group 0 Commit																														
dsa-MQSign	Working Group 0 Commit																														
dsa-NCCSign	Working Group 0 Commit																														
dsa-Peregrine	Working Group 0 Commit																														
pke-IPCC	Working Group 0 Commit																														
pke-NTRUplus	Working Group 0 Commit																														
pke-SMAUG_revised	Working Group 0 Commit																														
pke-TiGER	Working Group 0 Commit																														

Q & A