Autoreferat

1 Imię i nazwisko

Maciej Łukasz Obremski

2 Dyplomy

- **2013, doktorat z matematyki (informatyka teoretyczna)**

  Wydział Matematyki, Informatyki i Mechaniki, Uniwersytetu Warszawskiego,
  Tytuł: Flexible Two-Source Extractors and their Applications
  Opiekun: Prof. Stefan Dziembowski

- **2009, magisterium z matematyki**

  Wydział Matematyki, Informatyki i Mechaniki, Uniwersytetu Warszawskiego,
  Tytuł: Extreme Points Methods and Their Application to Abstract Bochner Theorem Proof
  Opiekun: Prof. Stanisław Kwapień

3 Zatrudnienie

- Maj 2020-
  Centre for Quantum Technologies (CQT), National University of Singapore
  Senior Research Fellow (equiv. Research Assist. Prof.)
- Październik 2018- Maj 2020
  CQT, National University of Singapore
  Research Fellow (equiv. PostDoc)
- Październik 2015- Wrzesień 2018
  Crypto group, Aarhus University
  Postdoc
- Październik 2013- Wrzesień 2015
  Wydział MIM UW (Wydział Matematyki, Informatyki i Mechaniki, Uniwersytetu Warszawskiego)
  Asystent
- Wrzesień 2013- Wrzesień 2015
  Cryptography Group, MIM UW
  Postdoc
- Październik 2011- Wrzesień 2013
  Wydział MIM UW
  Asystent naukowy

4 Omówienie osiągnięć, o których mowa w art. 219 ust. 1 pkt. 2 Ustawy.

4.1 Lista prac tworząca główny cykl

- **Leakage-resilient non-malleable codes.**
  D. Aggarwal, S. Dziembowski, T. Kazana, M. Obremski [1]
  Theory of Cryptography Conference (TCC) 2015
  CORE- Rank A, punkty MNiSW: 140

- **Non-malleable Reductions and Applications.**
  D. Aggarwal, Y. Dodis, T. Kazana, M. Obremski
  Symposium on Theory of Computing (STOC) 2015
  CORE- Rank A*, punkty MNiSW: 200

- **Inception makes non-malleable codes stronger.**
  D. Aggarwal, T. Kazana, M. Obremski
  Theory of Cryptography Conference (TCC) 2017
  CORE- Rank A, punkty MNiSW: 140

- **Continuous NMC Secure Against Permutations and Overwrites, with Applications to CCA Secure Commitments.**
  I. Damgård, T. Kazana, M. Obremski, V. Raj, L. Siniscalchi
  Theory of Cryptography Conference (TCC) 2018
  CORE- Rank A, punkty MNiSW: 140

- **Continuous non-malleable codes in the 8-split-state model.**
  D. Aggarwal, N. Dottling, J.B. Nielsen, M. Obremski, E. Purwanto
  Annual International Conference on the Theory and Applications of Cryptographic Techniques (Eurocrypt) 2019
  CORE- Rank A*, punkty MNiSW: 200

- **Stronger leakage-resilient and non-malleable secret-sharing schemes for general access structures.**
  D. Aggarwal, I. Damgard, J.B. Nielsen, M. Obremski, E. Purwanto, J. Ribeiro, M. Simkin
  Annual International Cryptology Conference (CRYPTO) 2019
  CORE- Rank A*, punkty MNiSW: 200

- **A constant rate non-malleable code in the split-state model.**
  D. Aggarwal, M. Obremski
  Foundations of Computer Science (FOCS) 2020
  CORE- Rank A*, punkty MNiSW: 200

4.2 Wstęp do kryptografii odpornej na wycieki i manipulacje danych

Przez większość historii świata kryptografia była jedynie zestawem sztuczek pozwalających względnie bezpiecznie przesłać tajne rozkazy. Przypominała bardziej występy magika ze swoimi ukrytymi kieszeniami i podwójnym dnem w skrzyniach, czy hollywoodzki film o Wschodnim Berlinie, a nie ścisłą naukę, którą jest dziś. Ciągły wyścig zbrojeń między ludźmi chcącymi ukryć wiadomość, a ludźmi chcącymi ją odczytać miał swą kulminację w pełnej, ścisłej formalizacji definicji bezpieczeństwa i ich rygorystycznej analizie.

[1] W dziedzinie, którą się zajmuję autorzy są zawsze wymieniani w kolejności alfabetycznej.

Stwierdzenia, "by złamać mój szyfr, trzeba być naprawde bardzo, bardzo sprytnym" zamieniliśmy na ścisłe redukcje jak: "by złamać ten szyfr, przeciwnik musiałby rozwiązać jakiś problem, którego nie umiemy rozwiązać od 50 lat". W końcu połączyliśmy bezpieczeństwo protokołów z problemami obliczeniowymi (co do których istnieją pewne dowody, że są trudne) lub z problemami natury teorioinformacyjnej, np. żeby złamać schemat przeciwnik musiałby zgadnąć cały tajny klucz, co jest niemożliwe, bo klucz jest długi i wylosowany jednostajnie (jak w one-time pad).

Gdy wydawało się, że idzie nam świetnie, nadeszła nowa fala pomysłów i ataków. Ataki te nie łamały matematycznych podstaw kryptografii - te były zbyt silne. Zamiast tego nowa fala ataków była popisem myślenia "outside the box". Zaczęto wykorzystywać słabości (i nieprzemyślane optymalizacje) implementacji lub po prostu zaczęto robić rzeczy, które nie były przewidziane w ścisłych modelach bezpieczeństwa. Wcześniej patrzyliśmy na urządzenie kryptograficzne jak na zamkniętą i odizolowaną czarną skrzynkę ("blackbox"), w środku był ukryty tajny klucz, zaś przeciwnik miał dostęp jedynie przez ściśle kontrolowany interfejs wejścia/wyjścia. Ale żadne fizyczne urządzenie nie jest izolowaną czarną skrzynką, musi czerpać energię elektryczną, wydziela promieniowanie elektromagnetyczne, wydziela ciepło, prędkość obliczeń może zależeć od tajnego klucza. Te wszystkie zmienne nie były wcześniej uwzględniane w wygodnym modelu czarnej skrzynki. W ten sposób nastała era *ataków pasywnych*, teraz przeciwnik mógł czerpać dane poboczne (ang. sidechannel information) i na ich podstawie próbować łamać bezpieczeństwo urządzenia kryptograficznego (e.g. [ECR]). Owe dane poboczne nazywamy zwykle *wyciekiem* (ang. leakage), a przeciwnika, który z nich korzysta, *pasywnym przeciwnikiem* (ang. passive adversary).

Gdyby tego było mało, pojawiły się pomysły idące dużo dalej i pojawili się *przeciwnicy aktywni* (ang. active adversary). Taki przeciwnik nie tylko pasywnie obserwuje urządzenie kryptograficzne, ale aktywnie w nim przeszkadza, złośliwie modyfikuje część urządzenia (na przykład zmieniając wartości na bramkach logicznych) (e.g. [AK96]) lub podszywając się pod innych uczestników protokołu.

Wspomniane ataki doprowadziły do powstania nowego trendu w teoretycznej kryptografii zapoczątkowanego przez [MR04, ISW03, IPSW06]. Celem było tworzenie protokołów i schematów, które już w modelu bezpieczeństwa uwzględniałyby zarówno pasywnych jak i aktywnych przeciwników. Ścisła analiza matematyczna gwarantowałaby bezpieczeństwo takich schematów i protokołów. Jedyne czego potrzebowaliśmy to dobre modele opisujące jak najszersze klasy ataków. W ciągu ostatnich lat zaproponowano wiele takich modeli (e.g. [ISW03, IPSW06, DP08, AGV09, NS09, BKKV10, DHLAW10, FPV11]).

W przypadku *ataków pasywnych* te modele uchwyciły większość, jeśli nie wszystkie, ataki możliwe w rzeczywistości. Ponadto, znane są liczne konstrukcje schmatów kryptograficznych bezpiecznych w tych modelach włączając w to generalny kompilator [GR12].

W przypadku *ataków aktywnych* sytuacja jest dużo mniej optymistyczna. Często wymagamy dodatkowych założeń, jak na przykład, że istnieje mały fragment urządzenia odporny na manipulacje (ang. tampering) [GLM+03], lub ograniczamy się do małej podklasy ataków jak probing czy ataki liniowe ([IPSW06, DSK12, Wee12]).

Tematem cyklu prac jest przeciwdziałanie *atakom aktywnym* zwanym też tampering attacks.

4.3 Wstęp: korekcja, detekcja i niekowalność

4.3.1 Wstęp do korekcji: rozgrzewka z kodami korekcji błędów

Kody korekcji błędów to jedne z najbardziej podstawowych obiektów w teorii kodów. Pozwalają na zakodowanie wiadomości m w słowo kodowe c o takiej właściwości, że nawet jeśli c zostanie "zaburzone"[2] to wciąż z tego zaburzonego słowa kodowego będziemy w stanie odczytać oryginalną wiadomość. To zaburzenie nie może być dowolne, łatwo sobie wyobrazić zaburzenie, które całkowicie kasuje słowo kodowe c i nadpisuje na nie ustalone słowo c', w takim wypadku nie ma szans na odczytanie oryginalnej wiadomości. Kody korekcji błędów zostały pomyślane w kontekscie zaszumionych kanałów (ang. noisy channels), więc "zaburzenie" reprezentuje tutaj losowy szum kanału. Formalnie mówiąc oczekujemy, że dla każdego

[2]Umyślnie nie używam słowa "zmanipulowane" (ang. tampered) by podkreślić losową naturę zaburzenia.

wektora v o małej wadze Hamminga (mała liczba niezerowych współrzędnych) będziemy w stanie odzyskać oryginalną wiadomość ze słowa kodowego $c' = c+v$. Ponieważ procedura odzyskiwania wiadomości zawsze się musi powieść (o ile v ma dostatecznie małą wagę Hamminga) można łatwo zauważyć, że $v(c)$ może zależeć od c i nadal będziemy w stanie odtworzyć wiadomość z $c' = c + v(c)$.

Niestety nie znamy naturalnego sposobu, by ograniczyć przeciwnika tylko do dodawania wektorów o małej wadze i ten model aktywnego przeciwnika nie jest realistyczny. Ponadto, nawet jeśli przeciwnik nie widzi c, tworząc wektor v o dowolnej wadze, nie jesteśmy w stanie trywialnie zagwarantować bezpieczeństwa. Wyobraźmy sobie prosty schemat kodowania: by zakodować 0 wysyłamy 00000...000, by zakodować 1 wysyłamy 11111...111. Przeciwnik bez znajomości słowa kodowego jest w stanie wyprodukować wektor $v = 11111...111$, po dodaniu tego wektora wiadomość zmieni się z 0 na 1 a z 1 na 0.

Question 4.1. *Czy możemy pozwolić przeciwnikowi dodawać wektor dowolnej wagi?*

4.3.2 Wstęp do detekcji: kody detekcji manipulacji algebraiczynych (AMD codes)

Dzięki kodom detekcji manipulacji algebraicznych (ang. algebraic manipulation detection codes) wprowadzonym w [CDF+08], odpowiedź na powyższe pytanie jest pozytywna.

Wiadomość $m \in \{0,1\}^{n \cdot k}$ możemy zinterpretować jako wielomian p nad ciałem $GF(2^n)$. Ściśle mówiąc, zapisujemy $m = (m_1, ..., m_k)$, gdzie każde $m_i \in GF(2^n)$ i wielomian p przyjmuje następującą postać

$$p_m(x) = x^{k+2} + \sum_{i=1}^{k} m_i x^i.$$

Mając wiadomość m, wybieramy jednostajnie losowe $x \in GF(2^n)$ i kodujemy m jako

$$m||x||p_m(x).$$

Gdzie $||$ oznacza konkatenację. Oględnie mówiąc bezpieczeństwo tego kodu można zredukować do faktu, że przeciwnik nie zna losowej wartości x gdy tworzy wektor v. Można pokazać, że aby dodać $v \neq 0$ i uzyskać dopuszczalne słowo kodowe (nie prowadzące do błędu dekodera) przeciwnik musiałby zgadnąć x, co jest informacyjnie teoretycznie niemożliwe[3].

Ahmadi i Safavi-Naini [AS13], a później Lin, Safavi-Naini, i Wang [LSW16] podali konstrukcje kodów AMD odpornych na wycieki (ang. leakage resilient AMD codes), w tym modelu przeciwnik może otrzymać/wyciec odrobinę informacji o słowie kodowym i na podstawie tej informacji wybrać odpowiednie $v(I)$, schematy [AS13, LSW16] pozostają bezpieczne nawet w tym scenariuszu. Tutaj przeciwnik najpierw wykonuje atak pasywny wyciekając jakieś informacje o stanie urządzenia, a następnie dokonuje ataku aktywnego preparując odpowiedni wektor v.

W jednej z późniejszych moich prac (która nie jest częścią głównego cyklu) [AKO18] pokazaliśmy optymalne konwersje parametrów (ang. trade-offs) pomiędzy code rate (stosunek długości wiadomości do długości słowa kodowego), a ilością informacji, którą przeciwnik może wyciec, by schemat pozostał bezpieczny.

Należy tutaj podkreślić, że przeciwnik, tworzący wektor v, nie może posiadać pełnej wiedzy o c, w takiej sytuacji jakiekolwiek bezpieczeństwo nie jest możliwe. Bardzo istotne jest też, że ten model jest kompletnie niezdolny do radzenia sobie z atakami, które kasują i nadpisują dane. Takie ataki wydają się być najbardziej podstawowymi i najprostszymi do wykonania - każdy, kto zostawił dyskietkę w sąsiedztwie magnesu, doskonale się o tym przekonał. Być może nadpisywanie danych nie wydaje się być najlepszym atakiem, ale należy je traktować jako swoiste "sanity check"- każdy rozsądny model ataków aktywnych musi pozwalać na kasowanie danych[4]. W modelu manipulacji algebraicznej, by nadpisać słowo c słowem

[3] Albo bardzo nieprawdopodobne, w praktyce mówimy o zdarzeniach o prawdopodobieństwie mniejszym niż 2^{-80}.

[4] W dalszej części pokażemy scenariusze, w których częściowe nadpisywanie danych jest możliwym atakiem prowadzącym do dowodu nieistnienia pewnych obiektów.

$c' = \text{const.}$, musielibyśmy dodać wektor $v = c' - c$, taki wektor niestety zależałby od całego c, a to, jak już wpominaliśmy, nie jest możliwe w modelu AMD

Question 4.2. *Czy istnieje model przeciwników aktywnych, który dopuszczałby szeroką klasę ataków aktywnych włączając w nią nadpisywanie i kasowanie danych?*

4.3.3 Wstęp do niekowalności: gdy korekcja i detekcja nie są możliwe

Korekcja i detekcja nie zawsze mogą być możliwe (na przykład poprzez nadpisanie danych). Motywowani tym Dziembowski, Pietrzak i Wichs [DPW10] zaczęli rozważać nowy obiekt - kody niekowalne (ang. non-malleable codes (NMC)). Jest to uogólnienie kodów detekcji i korekcji zbudowane w oparciu o koncepcje *niekowalności* (ang. non-malleability) wprowadzonej przez [DDN00].

Model jest bardzo naturalny i elegancki. Mając wiadomość m, zakodujemy ją do $\mathsf{Enc}(m) = c$, słowo kodowe jest składowane na urządzeniu, przeciwnik wybiera dowolne $f \in \mathcal{F}$, i słowo kodowe jest manipulowane (ang. tampered to) do $c' = f(c)$, w końcu dekodujemy nowe słowo kodowe $Dec(f(c)) = m'$. W wariancie *korekcji* chcieliśmy by $m' = m$, w wariancie *detekcji* chcialiśmy by $m' = m$ lub $m' = \perp$ (gdzie $\perp$ jest specjalnym symbolem oznaczającym błąd dekodera). Jak już ustaliliśmy wcześniej, jeśli rodzina $\mathcal{F}$ zawiera funkcje stałe, to ani korekcja, ani detekcja nie są możliwe. Istnieje jednak naturalna definicja, która wciąż daje istotne gwarancje bezpieczeństwa. Kody niekowalne przeciwko $\mathcal{F}$ gwarantują, że po opisanej powyżej manipulacji albo $m' = m$, albo m' jest kompletnie niezależne od m, w szczególności $m' = m + 1$ nie jest możliwe.

Żeby uzasadnić taką definicję musimy wspomnieć o *atakach skorelowanym kluczem* (ang. related key attacks) (e.g. [Bih94, BK03]). Tutaj przeciwnik manipuluje urządzeniem $D(K)$ zawierającym w sobie tajny klucz K. Przeciwnik jest w stanie uzyskac dostęp do wejścia/wyjścia urządzenia $(D(K'))$ gdzie K' jest skorelowanym kluczem (np. poprzez manipulacje urządzenia $D(K)$), często jest to wystarczające do złamania bezpieczeństwa oryginalnej maszyny. Należy podkreślić, że gdyby klucz był zabezpieczony przez kod niekowalny, wtedy K' byłoby kompletnie niezależne od K i taki atak by nic nie dał.

Niekowalność nie jest ograniczona tylko do kodów, można ją rozważać w kontekscie wielu innych obiektów na przykład *schematów podziału sekretu (secret sharing), seeded extractors, ekstraktorów dwuźrodłowych (two source extractors), schematów zobowiązań (commitments)* etc. We wszystkich powyższych przypadkach idea jest dość podobna, albo oryginalna wiadomość lub oryginalne wyjście zostanie zachowane, albo zostanie całkowicie zniszczone.

4.4 Kody niekowalne (NMC)

4.4.1 Popularne klasy manipulacji

Powiedzieliśmy wcześniej, że uzyskanie $m' = m + 1$ nie może być możliwe przy użyciu NMC. To oznacza, że klasa manipulacji $\mathcal{F}$ musi być ograniczona. W szczególności następująca funkcja nie może należeć do $\mathcal{F}$:

$$f(c) = \mathsf{Enc}(\mathsf{Dec}(c) + 1).$$

Najbardziej popularne klasy manipulacji to:

- **Independent bit tampering** [DPW10]- każdy z bitów słowa kodowego jest manipulowany niezależnie, tj. przeciwnik decyduje czy zachować, odwrócić, ustawić na 0 lub na 1 każdą pozycję indywidualnie
- **Bounded depth circuits** [BDSG+18]- przeciwnik może manipulować słowem kodowym z pomocą dowolnego układu logicznego o ograniczonej głębokości
- **Bounded locality** [BDSKM16]- przeciwnik może manipulować z użyciem funkcji o ograniczonej lokalności tj. każdy bit nowego słowa kodowego może zależeć tylko od ograniczonej liczby bitów oryginalnego słowa c

- **Bounded degree polynomial or bounded polynomial time tampering** [BCL+20, BDSK+19]
- **Permutations** [AGM+14, AGM+15] [DKO+18]- przeciwnik mozże dowolnie permutować bity słowa kodowego, niektóre model dopuszczają nadpisywanie, inne dopuszczają ograniczoną liczbę negacji
- **Split state tampering** [DKO13][CG14, CZ14, ADL14][ADKO15a, ADKO15b][CGL16, Li17, KOS17] [AKO17][GMW18, KOS18, Li19][AO20]- najbardziej popularny z modeli, opiszemy go szczegółowo poniżej

4.4.2 Manipulacje split state: co to takiego i dlaczego jest to istotne

Jednym z najbardziej popularnych modeli manipulacji jest model $t-$*split state*. W tym modelu słowo kodowe jest podzielone pomiędzy t *części*, zaś każda z nich jest manipulowana niezależnie. Formalnie mówiąc, niech $(X_1, ..., X_t) \in \mathcal{X}_1 \times ... \times \mathcal{X}_t$ oznacza owe t części (ang. *states*) słowa kodowego, przeciwnik wybiera t funkcji $f_1, ..., f_t$, gdzie $f_i : \mathcal{X}_i \to \mathcal{X}_i$, a następnie manipuluje z ich pomocą słowem kodowym: $(f_1(X_1), ..., f_t(X_t))$. Możemy myśleć o tym, jak o t przeciwnikach, każdy manipuluje jedną częścią. Przeciwnicy ci umówili się wcześniej co do strategii, ale podczas samej manipulacji nie mają możliwości porozumiewać się ze sobą. Więc i'ty przeciwnik nie zna niczego więcej niż tylko części X_i.

Jest to bardzo szeroka klasa manipulacji, gdyż żadna z funkcji f_i nie jest w żaden sposób ograniczona obliczeniowo, jedynym ograniczeniem jest to, że części są manipulowane osobno. Ponieważ ten model nie zakłada żadnych dodatkowych ograniczeń na manipulacje, obejmuje on dosłownie wszystkie ataki, o ile tylko uda nam się utrzymać części słowa kodowego oddzielone od siebie.

Dlaczego split state jest interesujący? Po pierwsze, jak już wspominaliśmy, jest to bardzo szeroka klasa funckji manipulacji. Po drugie są scenariusze kryptograficzne, gdzie utrzymanie oddzielności części pamięci jest bardzo naturalne- na przykład schematy dzielenia sekretu (ang. secret sharing) czy obliczenia wielopodmiotowe (ang. multiparty computation). Po trzecie, i w mojej opinii najważniejsze, model ten ma wiele zastosowań jako techniczne narzędzie. Kody niekowalne w split state (a zwłaszcza w $2-$split state) mają liczne zastosowania, przede wszystkim w konstrukcjach:

- Kodów niekowalnych w innych bardziej naturalnych modelach:

  Small depth circuits [BDSG+18]

  Local tampering [BDSKM16]

  Decision Trees [BGW19]

  Continuous NMCs [ADN+19b]
- Niekowalnych schematach dzielenia sekretu (ang. non-malleable secret sharing) [GK18a, BS18][ADN+19a]
- Privacy amplification (PA) [CKOS19]
- Niekowalnych schematach zobowiązań (ang. non-malleable committments) [GPR16]

4.5 Wyścig po constant-rate[5] w modelu 2 split state

Przez parę lat od publikacji [DPW10] poczyniono niewielkie postępy w konstrukcji informacyjno teoretycznie[6] bezpiecznych kodów niekowalnych. Wtedy nadeszła seria przełomowych wyników: [DKO13] [ADL14, CZ14] [ADKO15a].

[5] Constant rate odnosi się do stosunku długości wiadomości do długości słowa kodowego, innymi słowy chcemy by długość słowa kodowego była liniową funkcją długości wiadomości. Ma to kolosalne znaczenie dla wydajności zastosowań.

[6] Przez bezpieczeństwo teorio-informacyjne rozumiemy bezpieczeństwo nie polegające na żadnych założeniach obliczeniowych. Podczas gdy wiele klas manipulacji polega na separacjach między różnymi klasami obliczeniowymi, model split state nie wymaga żadnych takich założeń i pozostanie bezpieczny nawet jeśli P=NP.

Podkreślam tutaj, że [DKO13] nie jest częścią głównego cyklu, gdyż została opublikowany przed moim doktoratem.

- Pierwszym przełomem była [DKO13](CRYPTO), zbudowaliśmy pierwszą konstrukcję NMC w modelu 2−split state, chociaż tylko dla jednobitowych wiadomości. Pokazaliśmy, jak używać elastycznych ekstraktorów dwu-źródłowych[7] do budowy kodów niekowalnych. Zaprezentowaliśmy serię nowych technik, najważniejszą wśród nich był *domain partition argument*. Pokazaliśmy, jak dzielić przestrzeń wszystkich słów kodowych na podzbiory, argumentować bezpieczeństwo na każdym podzbiorze z osobna, a następnie automatycznie dowodzić bezpieczeństwa całości, łącząc wszystkie fragmenty dziedziny. Ta technika jest szeroko stosowana do dziś.

  Natura kodów niekowalnych wymusza na nas używanie argumentów entropijnych, by móc aplikować lematy z teorii ekstraktorów i wycieków. To oznacza, że musimy podzielić możliwe manipulacje na te, które zachowują istotną korelację między wejściem, a wyjściem funkcji manipulacyjnej (na przykład bijekcje), oraz te manipulacje, które gubią tę korelację (na przykład funkcje stałe). To ładnie działa dopóki funkcja jest bijekcją albo jest stała, ale zdecydowana wiekszość funkcji taka nie jest. W tej pracy wprowadziliśmy prostą ale niesamowicie pożyteczną technikę: popatrzmy na $B = \{x \mid |f^{-1}(x)| \text{ jest mały}\}$, oto fragment dziedziny, na ktorym funkcja bedzie zachowywała sie jak bijekcja a wyjście zachowa entropię wejścia (jeśli X ma wysoką min-entropię, wtedy $f(X)|X \in B$ też ma wysoką min-entropię). Tak się dzieje, jeśli tylko B nie jest zbyt mały, ale jeśli B jest na prawdę mały (by powstrzymać transfer entropi musi być mniejszy niż zaniedbywalny (negligible) fragment dziedziny) wtedy nie musimy się martwić, bo szansa, że $X \in B$ jest zaniedbywalna. W oparciu to tego typu argumenty byliśmy w stanie zastosować lematy z teorii ekstrakcji a następnie zbudować pierwszy kod niekowalny w modelu 2 split state. Sformuowaliśmy również hipotezę, że dla dowolnych funkcji manipulacji f, g działających na iloczynie skalarnym (dot product), $\langle f(X), g(Y)\rangle$ jest randomizowaną funkcją afiniczną orginalnego iloczynu skalnarnego $\langle X, Y\rangle$.

- Drugi przełom nadszedł z [ADL14](STOC). Autorzy, budując na technikach wprowadzonych powyżej, udowodnili naszą hipotezę. Dowód przebiega przez sprytne połączenie problemu z twierdzeniem quasi-polynomial Freiman-Ruzsa (QPFR). Następnie, używając udowodnionej hipotezy, autorzy zbudowali kod niekowalny w modelu 2−split state dla wiadomości dowolnej długości. Niestety główny atut tej pracy stał się też jej głównym minusem, addytywna kombinatoryka wymusiła astronomiczny rate of the code[8]. Dla wiadomości złożonej z n bitów, słowo kodowe ma długość ponad $O(n^7)$ bitów, później poprawiono na wciąż niesatysfakcjonujące n^5 bitów. I tak rozpoczął się wyścig o NMC z contant rate w modelu 2 split state.

- W [CZ14](FOCS) autorzy eksporują nowe techniki budowy kodów niekowalnych. Wprowadzają nowy obiekt: niekowalny ekstraktor wielo-źródłowy (obiekt, który gwarantuje, że modyfikacja wejścia "re-losuje" wyjście) i na jego podstawie budują kod niekowalny w 9 split state o constant rate. Jest rzeczą jasną, że im mniej stanów tym konstrukcja jest trudniejsza, więc NMC w modelu 2 state z constant-rate pozostawało otwartym problemem.

- Wtedy opublikowaliśmy [ADKO15a](STOC). Wprowadziliśmy nowe pojęcie: niekowalną redukcję (non-malleable reduction). Sposób na budowanie na sobie wielu poziomów kodu, tak by powoli redukować moc przeciwnika, tak długo, aż osiągniemy pełną niekowalność[9]. W tej pracy pokazaliśmy jak kompilować konstrukcje w 9−split state [CZ14] do konstrukcji bezpiecznej w modelu 2 split state.

[7] Funkcja, która przetwarza dwa niezależne źródła losowości o pewnej min-entropii w jednostajnie losowe wyjście. Formalnie, jeśli $\mathbf{H}_\infty(X) + \mathbf{H}_\infty(Y)$ przekracza pewien próg, oraz X, Y są niezależne, wówczas $\mathsf{Ext}(X, Y)$ jest niemal (ang. negligibly close) jednostajne. Gdzie min-entropie definiujemy jako $\mathbf{H}_\infty(X) = -\log \max_x P(X = x)$. Przymiotnik ęlastycznyóznacza, że wymagany próg entropii jest ustalony dla sumy entropi źródeł a nie dla każdego źródła z osobna.

[8] stosunek długości wiadomości do długości słowa kodowego

[9] Do pojęcia niekowalnych redukcji wrócimy w późniejszej sekcji gdzie omówimy je dokładniej.

To wszystko zachowując constant-rate konstrukcji [CZ14]. Niestety radość była przedwczesna, 2 lata później [Li17] odkrył błąd w jednym z dowodów, konstrukcja zdawała się pozostawać bezpieczną, ale nikt nie wiedział, jak tego dowieść. Rezultaty z tej pracy nie załamały się kompletnie, pozostał częściowy wynik z konstrukcją w 5 split state z kwardratową długością słowa kodowego[10] oraz liczne redukcje nie składające się jednak do pełnego wyniku.

Nie wiedząc jeszcze, że wyścig po constant-rate NMC w 2 split state pozostaje otwarty, zajęliśmy się silniejszymi modelami niekowalności.

- W [ADKO15b](TCC) rozszerzyliśmy model split state i dopuściliśmy pewną ograniczoną komunikację między stanami. Formalnie mówiąc oryginalne słowo kodowe (X_1, X_2) było manipulowane do $(f(X_1, h_1(X_2)), g(X_2, h_2(X_1)))$ gdzie h_1, h_2 są dowolnymi funkcjami o ograniczonym rozmiarze obrazu (ten model odwzorowuje ograniczoną ilość bitów komunikacji pomiędzy stanami). Pokazaliśmy ogólny kompilator, który bierze dowolny 2 split state NMC i zamienia go w kod niekowalny odpory na wycieki. Odporność na wycieki oznacza nic innego jak dopuszczenie ograniczonej komunikacji między stanami.

- W [AKO17](TCC) położyliśmy podstawy pod wiekszość naszych przyszłych prac w tej dziedzinie. Pokazaliśmy jak uzyskać pierwszy *silny* (ang. strong) i pierwszy *super-silny* (ang. super-strong) kod niekowalny[11] w modelu 2 split state. By opowiedzieć o wynikach musimy najpierw cofnąć się do definicji kodów niekowalnych. Kodujemy wiamość jako $\mathsf{Enc}(m) = (X_1, X_2)$, przeciwnik tworzy $\mathsf{Dec}(f(X_1), g(X_2)) = m'$, oczekujemy, że albo $m = m'$ albo m i m' są niezależne. Wariant *silny* wymaga, że jeśli tylko słowo kodowe zostało zmienione $(f(X_1), g(X_2)) \neq (X_1, X_2)$ to m i m' muszą być niezależne, innymi słowy przypadek $m = m'$ może się zdarzyć tylko jeśli manipulacja nie nastąpiła. W *super-silnym* wariancie gwarantujemy, że jeśli manipulacja nastąpiła, to nie tylko m' będzie niezależne od m, ale całe słowo kodowe $(f(X_1), g(X_2))$ będzie niezależne od m. Innymi słowy gwarantujemy całkowite zniszczenie, nowe słowo kodowe nie wygeneruje błędu dekodera, tylko jeśli przeciwnik naprawdę je prawie kompletnie skasuje w stopniu uniemożliwiającym odzyskanie oryginalnej wiadomości.

  Uzyskaliśmy ten wynik, wprowadzając nową technikę zwaną *inception coding*. Wprowadza ona pewną cykliczność: słowo kodowe koduje wiadomość, ale wiadomość koduje pewną informację (validity checks) o słowie kodowym. Pokazujemy, jak wydajnie uzyskiwać tę cykliczność, używając specjalnych testów (ang. checks), które skonstruowaliśmy na tę okazję. Popatrzmy pobierznie na kompilator budujący silne NMC przeciw 2 split state ze zwykłego NMC przeciw 2 split state. Gdy już wiemy, jak uzyskać cykliczność, reszta jest intuicyjnie dość prosta (choć po drodze są liczne przeszkody techniczne, np. należy pokazać, że wprowadznie cykliczności nie zaburzy niekowalności). Wystarczy zauważyć, że jedynym przypadkiem, którym musimy się zająć jest następujący przypadek: wiadomość pozostała niezmieniona $m' = m$, ale słowo kodowe zostało zmanipulowane. Jeśli w tym przypadku bylibyśmy w stanie zmusić dekoder do zwrócenia błędu, to właśnie zamieniliśmy NMC w silne NMC. Tutaj w grę wchodzi cykliczność: m koduje pewne testy na słowe kodowym, słowo kodowe się zmieniło, ale testy pozostały takie same (ponieważ m'=m), z prawdopodobieństwem niemal 1 (minus zaniedbywalny czynnik) nowe słowo kodowe nie spełnia testów zakodowanych w wiadomości, więc dekoder zwróci błąd.

  Jeden z technicznych problemów wymagał od nas pokazania, że KAŻDY NMC w 2 split state jest też odporny na wycieki w trochę słabszym wariancie niż ten rozważany w [ADKO15b]. Później, w

[10] Dla n bitowej wiadomości długość słowa kodowego wynosi $O(n^2)$

[11] Pojawiły się liczne żarty, że dobór nazw odzwierciedla naszą megalomanię. Śpieszę wyjaśnić, że wbrew obiegowej opinii to nie my wymyśliliśmy te nazwy. Silne NMC były wprowadzone już w pierwszej pracy [DPW10], historia super-silnych NMC jest nieco bardziej złożona, my przypisujemy ten obiekt autorom [FMNV14], nasza definicja jest jedynie wyekstrachowaną i bardziej usystematyznowań wersją wcześniejszych definicji, zaś nazwa super-strong jest połączeniem strong od [DPW10] z super od [FMNV14].

[BFO+20] pokazaliśmy, używając powyższych technik, że KAŻDY kod niekowalny w 2 split state jest odporny na wycieki (tj. dopuszcza ograniczoną komunikację między stanami) według definicji [ADKO15b].

Ta praca pokazuje również, że super-silne NMC w modelu 2 split state są też *persistent* continuous NMC. Ten wynik omówimy w sekcji poświęconej ciągłym manipulacjom (ang. continuous tampering).

Po tym jak wyszedł na jaw błąd w [ADKO15a] wznowiono wyścig o constant-rate. W pracach [Li17, Li19] autorzy udoskonalali konstrukcje niekowalnych ekstraktorów dwu-źródłowych wprowadzonych przez [CG14, CGL16]. Najnowszy wynik [Li19] osiągnął konstrukcję z rate $\frac{c \cdot \log\log\log 1/\varepsilon}{\log\log 1/\varepsilon}$ dla pewnej stałej c. Ta konstrukcja osiąga constant-rate, gdy ε jest stały, ale rate zbiega do 0 jeśli ε jest zaniedbywalny względem n, zaniedbywalny ε jest absolutnie konieczny, więc i ten wynik nie rozwiązywał problemu.

Wiele prac poszło w innym kierunku, poprzez konstrukcje niekowalnych koderów losowości ([KOS18, KOS17]). W tej wersji nie możemy wybrać konkretnej wiadomości do zakodowania, zamiast tego kodowana jest jednostajnie losowa wiadomość. Przejście z losowej wiadomości do konkretnej wiadomości wydaje się może małą przeszkodą, jednak problem *wydajnego samplowania z przeciwobrazu* (ang. efficient preimage sampling)[12] jest wszechobecny w tej dziedzinie i często prowadzi do licznych komplikacji, np. w [CZ14] autorzy musieli dodać dodatkowy stan tylko po to, by zagwarantować wydajne kodowanie wybranej wiadomości.

Wraz z poprawiającym się rate coraz więcej prac używało niekowalnych kodów i ekstraktorów by budować nowe obiekty, jak na przykład niekowalne schematy zobowiązań (ang. non-malleable commitments) czy niekowalne schematy dzielenia sekretu (non-malleable secret sharing) [GPR16, GK18a, GK18b][BS18, SV18][ADN+19a] (wszystkie na konferencjach rangi A*).

Dla kompletności należy tutaj wspomnieć o równolegle rozwijającej się linii prac [CCFP11, CCP12, CKM11, FMVW14, AGM+14, AGM+15, BDSKM16, FHMV17, BDSKM18, BDSG+18][DKO+18], które albo używały albo budowały kody niekowalne dla innych rodzin manipulacji $\mathcal{F}$, ale nie koncentrowały się na modelu split state.

4.6 Huston, Orzeł się potknął, ale wylądował

Wreszcie w 2020 w [AO20](FOCS) podaliśmy pierwszą konstrukcję o constant rate i zaniedbywalnym błędzie w 2 split state model. Połączyliśmy wyniki i techniki z [ADL14] [ADKO15a, AKO17] i pokazaliśmy nowe techniki bootstrapping, które pozwoliły nam na uzyskanie wyniku. Ta praca wypełniła lukę pozostawioną 5 lat wcześniej przez błąd w [ADKO15a].

4.6.1 Redukcje niekowalne

By zrozumieć wynik, musimy ponownie wrócić do definicji kodów niekowalnych i wprowadzić pojęcie redukcji niekowalnej (wprowadzone w [ADKO15a]). W NMC parzymy na relacje pomiędzy m a $m' = f(\mathsf{Enc}(m))$ (dla pewnego $f \in \mathcal{F}$), i oczekujemy, że albo $m = m'$ albo m, m' są niezależne. Możemy na to spojrzeć z innej perspektywy: popatrzmy na m' jako na funkcję m, tj. $m' = g(m)$. Jeśli g jest (randomizowaną) kombinacją *identyczności* i *funkcji stałych* wtedy mamy kod niekowalny, identyczność reprezentuje przypadek $m' = m$, zaś funkcje stałę odpowiadają za m' niezależne od m. Możemy sobie wyobrazić, że zamiast identyczności i funkcji stałych dopuszczamy g by była dowolną randomizowną kombinacją funkcji z $\mathcal{G}$, to jest właśnie *redukcja niekowalna z $\mathcal{F}$ do $\mathcal{G}$*, zapisujemy jako: $\mathcal{F} \Rightarrow \mathcal{G}$. Siła tego pojęcia wypływa z dwóch obserwacji:

Observation 4.1. *Jeśli* $(\mathsf{Enc}, \mathsf{Dec})$ *jest redukcją* $\mathcal{F} \Rightarrow \mathsf{NM}$, *gdzie* NM *jest klasą zawierającą tylko identyczność i funkcje stałe, wtedy* $(\mathsf{Enc}, \mathsf{Dec})$ *jest kodem niekowalnym przeciw* $\mathcal{F}$.

[12] Proces kodowania jest często procesem samplowania ze zbioru słów kodowych kodujących konkretną wiadomość

Observation 4.2. *Jeśli* $(\mathsf{Enc}_1, \mathsf{Dec}_1)$ *jest redukcją* $\mathcal{F} \Rightarrow \mathcal{G}$ *oraz* $(\mathsf{Enc}_2, \mathsf{Dec}_2)$ *jest redukcją* $\mathcal{G} \Rightarrow \mathcal{H}$*, wtedy kompozycja schematów* $(\mathsf{Enc}_1 \circ \mathsf{Enc}_2, \mathsf{Dec}_2 \circ \mathsf{Dec}_1)$ *jest redukcją* $\mathcal{F} \Rightarrow \mathcal{H}$*.*

Powyższe obserwacje pozwalają nam na stopniowe redukowanie siły przeciwnika, aż w końcu w ostatnim kroku dotrzemy do klasy NM. Należy tutaj wspomnieć, że klasa $\mathcal{G}$ nie musi być podklasą $\mathcal{F}$, na przykład wiemy, jak zredukować następujące klasy do (sic!) klasy 2−split state tampering: układy logiczne o ograniczonej głębokości ([BDSG+18]), a local tampering ([BDSKM16]) czy drzewa decyzyjne ([BGW19]).

4.6.2 Brakująca redukcja

By wyjaśnić wyniki z [AO20], musimy najpierw wyjaśnić wyniki z [ADKO15a], a to wymaga wprowadzenia następujących klas:

- **2-Split state**: dyskutowaliśmy już o tej klasie oryginalne słowo kodowe (X_1, X_2), jest manipulowane z pomocą dowolnych f, g a wynikiem manipulacji jest $(f(X_1), g(X_2))$.
- 2−**Lookahead**: Ta klasa jest ograniczeniem powyższej klasy. Każda część jest podzielony na mniejsze fragmenty pamięci $X_1 = (X_1^1, X_1^2, X_1^3)$, oraz $X_2 = (X_2^1, X_2^2, X_2^3)$ (tutaj podzieliliśmy każdą część na 3 fragmenty pamięci, ale ten model dopuszcza dowolną stałą liczbę fragmentów). Manipulacja X_1 zachodzi niezależnie od X_2 (tak jak w split state) ale dodatkowo zachwana jest chronologiczna struktura manipulacji: najpierw manipulujemy X_1^1, potem manipulacja X_1^2 może zależeć od X_1^1, X_1^2 etc. Formalnie mówiąc przeciwnik wybiera $f_1, f_2, f_3, g_1, g_2, g_3$ a słowo kodowe po manipulacji wygląda następująco:

$$\left(\left(f_1(X_1^1), f_2(X_1^1, X_1^2), f_3(X_1^1, X_1^2, X_1^3)\right), \left(g_1(X_2^1), g_2(X_2^1, X_2^2), g_3(X_2^1, X_2^2, X_2^3)\right)\right).$$

- 2−**Lookahead and Forgetful** ta klasa pozwala na więcej niż powyższa. Słowo kodowe ma tę samą strukturę: $((X_1^1, X_1^2, X_1^3), (X_2^1, X_2^2, X_2^3))$, ale przeciwnik ma wybór, może albo manipulować z pomocą klasy lookahead (jak wyżej) albo z pomocą klasy forgetful. Jeśli zdecyduje się na tę drugą, to wybiera jeden fragment pamięci, który zostanie całkowicie usunięty, powiedzmy, że X_2^2. Dzięki uzyskanej w ten sposób przestrzeni przeciwnik może kompletnie złamać strukture split state i zmanipulowane słowo kodowe wygląda następująco:

$$h(X_1^1, X_1^2, X_1^3, X_2^1, X_2^3),$$

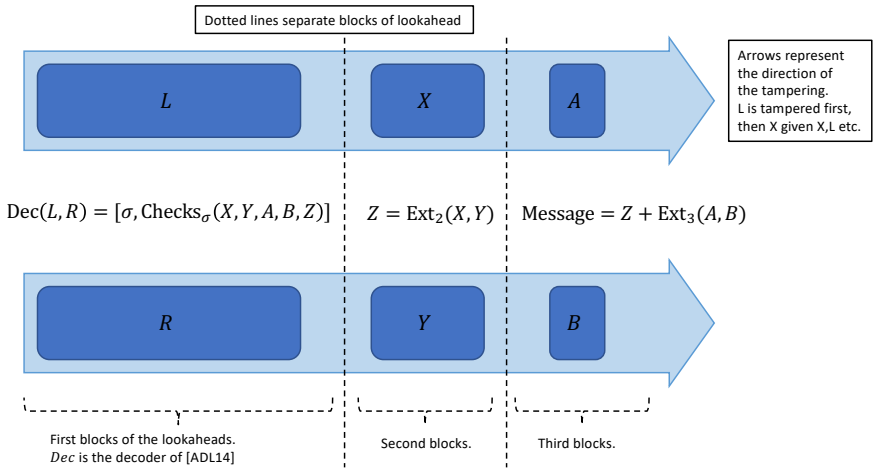
  dla dowolnej funkcji h. Podkreślamy, że fukcja h nie zachowuje ani struktury lookahead ani split state.

Teraz możemy wyjaśnić wyniki z [ADKO15a], praca składa się z dwóch głównych redukcji (każda ma pare stopni, ale to pomijamy):

1. Redukcja z 2−split state do 2−Lookahead and Forgetful
2. Redukcja z 2−Lookahead and Forgetful do NM

Razem te dwie redukcje dają kod niekowalny przeciwko manipulacjom w 2−split state. Co istotne, każda z tych redukcji zachowuje constant-rate, więc i kompozycja jest constant-rate. Lemat, w którym [Li17] znalazł błąd, był częścią reducji w kroku 2. Praca wciąż dawała konstrukcję w 5 split state z przyzwoitym (jak na tamte czasy) rate, ale konstrukcja z constant rate w 2 split state wydawała się poza zasięgiem.

Szanse na naprawienie brakującego dowodu były nikłe mimo, że był on intuicyjnie oczywisty. Niestety, jak dowodzi przypadek [DKO13] i [ADL14], ta dziedzina zdaje się być połączona z addytywną kombinatoryką, a tam nasze intuicje niewiele znaczą. W związku z tym w [AO20] postanowiliśmy zbudować redukcję z kroku 2 od zera.



Rysunek 1: Schemat kodowalnia kodu niekowalnego przeciwko manipulacjom 2−lookahead.

Wypracowaliśmy nową technikę bootstrapping, która pozwoliła nam użyć bezpiecznego kodu ([ADL14]), o bardzo słabym rate, do zakodowania jedynie bardzo krótkich testów (checks), owe testy miały rozmiar niezależny od rozmiaru wiadomości więc zły rate [ADL14] nie miał wpływu na rate całego kodu. Następnie użyliśmy serii argumentów ekstraktorowych połączonych z wspomnianymi wyżej testami, co umożliwiło nam dodanie dodatkowych własności niekowalności do standardowych ekstraktorów. Na rysunku 1 podajemy uproszczony schemat kodowania przeciwko manipulacjom 2−lookahead, ten schemat jest następnie kompilowany, by uzyskać dodatkowo odporność na manipulacje z klasy forgetful.

4.7 Continuous NMC

Jak już wspominałem wcześniej, badania nad constant-rate NMC przeplatały się z badaniami nad silniejszymi definicjami niekowalności. Dotąd rozważaliśmy tylko jednorazowe manipulacje, tj. słowo kodowe $\mathsf{Enc}(m)$ jest modyfikowane tylko raz z pomocą $f \in \mathcal{F}$ i zmanipulowane słowo kodowe to $f(\mathsf{Enc}(m))$. W [FMNV14] autorzy rozważają silniejsze wersje niekowalności, użyteczne na przykład do odpornych na manipulacje obliczeń na architekturze von Neumanna. Ten silniejszy wariant dopuszcza wielokrotne manipulacje: przeciwnik wybiera manipulację $f_1 \in \mathcal{F}$ i uzyskuje s $m'_1 = \mathsf{Dec}(f_1(\mathsf{Enc}(m)))$, w oparciu o ten wynik wybiera f_2 i uzyskuje $m'_2 = \mathsf{Dec}(f_1(\mathsf{Enc}(m)))$, i tak dalej, aż dekoder zwróci błąd $\mathsf{Dec}(f_n(\mathsf{Enc}(m))) = \perp$ tj. nastąpi detekcja. Powiemy, że kod jest continuous NMC jeśli cała seria $m'_1, ..., m'_n$ nie zdradza oryginalnej wiadomości. Należy poczynić tutaj dwie uwagi.

Remark 4.1. *Jeśli $f_i \equiv \mathsf{Id}$ wtedy $m'_i = m$, co kompetnie złamie własność wskazaną powyżej. W związku z tym, zawsze gdy $m'_i = m$ podmieniamy m'_i na specjalny symbol* same. *Intuicja za tym jest następująca: jeśli przeciwnik nie manipulował tajnego klucza to urządzenie pozostanie bezpieczne, bo nie nastąpił atak skorelowanym kluczem. Jednocześnie przeciwnik jest w stanie wywnioskować coś nietrywialnego z takiego ataku mianowicie, że słowo kodowe jest w zbiorze punktów stałych jego manipulacji. Dlatego przeciwnik otrzymuje informację zwrotną w postaci symbolu* same.

Remark 4.2. *Należy również zwrócić uwagę, że autodestrukcja (zatrzymanie manipulacji w momencie*

gdy nasąpił błąd dekodera, tj. nastąpiła detekcja) jest absolutnie konieczne. Kody niekowalne nie istnieją, jeśli przeciwnik może dalej manipulować mimo błędów dekodera. Dobrze ilustruje to następujący atak wymagający jedynie możliwości nadpisywania poszczególnych bitów. Niech $c = (c_1, ..., c_n)$ *będzie słowem kodowym, gdzie* $c_i \in \{0, 1\}$*. Przeciwnik nadpisuje pierwszy bit przez* 0*:* $(0, c_2, ..., c_n)$ *jeśli dekoder zwróci* same *to wiemy, że pierwszy bit słowa kodowego był* 0*, w przeciwnym wypadku pierwszy bit był* 1*, nastepnie powtarzamy proces dla drugiego bitu etc.. W końcu nauczymy się całego słowa kodowego i będziemy mogli odkodować oryginalną wiadomość, a co za tym idzie złamiemy bezpieczeństwo schematu.*

W [JW15] autorzy rozważają różne warianty definicji continuous NMC, najistotniejsze dwie werjse to *persistent* i *non-persistent* (lub *resettable*).

- **Resettable** lub **non-persistent** jest silniejszym wariantem. manipulacja odbywa się tak jak opisaliśmy wcześniej $m'_1 = f_1(c), ..., m'_n = f_n(c)$, gdzie $c = \mathsf{Enc}(m)$.
- W **Persistent** po manipulacji przeciwnik nie może resetować słowa kodowego do oryginalnej postaci, zamiast tego musi pracować z tym co zostało ze słowa po wcześniejszych manipulacjach: $m'_1 = f_1(c), m'_2 = f_2(f_1(c)), ..., m'_n = f_n(..(f_1(c)..)$. Ten wariant stara sie uchwycić ideę, że pewne manipulacje, zwłaszcza fizyczne, są nieodwracalne.

Co ciekawe, wersja *resettable* jest tak dużo silniejsza, że kody *resettable* mogą nie istnieć tam, gdzie kody *persistent* są jeszcze możliwe. Najistotniejszym wynikiem jest nieistnienie resettable continuous non-malleabe codes (CNMC) w modelu 2 split state. Naszkicujemy dowód ten faktu tutaj. Z właściwości kodów niekowalnych (dowód w [ADKO15b]) wiemy, że. istnieje trójka x, y, z taka, że $\perp \neq \mathsf{Dec}(x, z) \neq \mathsf{Dec}(y, z) \neq \perp$. Przeciwnik zastępuje drugą część słowa kodowego wektorem z, a następnie nadpisuje pierwszy stan z pomocą x lub y zależnie od tego czy pierwszy bit słowa kodowego jest 0 czy 1. W ten sposób przeciwnik nauczył się pierwszego bitu słowa kodowego, przeciwnik resetuje słowo kodowe i powtarza ten atak z kolejnymi bitami pierwszej części słowa kodowego. Kiedy pozna już wszystkie bity pierwszej części, to znajduje trójkę w, v, q taką, że $\perp \neq \mathsf{Dec}(w, v) \neq \mathsf{Dec}(w, q) \neq \perp$ i w ten sposób uczy się bitów drugiej części słowa kodowego. Przeciwnik w ten sposób poznał cale słowo kodowe (a co za tym idzie całą oryginalną wiadomość) bez ryzyka detekcji.

- W [AKO17] poza konstrukcjami silnych i super-silnych NMC pokazaliśmy także, że każdy super-silny NMC przeciwko modyfikacjom w 2 split state jest też *persistent* continuous non-malleable code (CNMC) przeciwko manipulacjom w 2−split state model. To była pierwsza informacyjnie teoretyczna konstrukcja w nietrywialnym modelu. Ponadto, biorąc pod uwagę nieistnienie kodów resettable w modelu 2 split state, otrzymujemy separacje pomiędzy persistent a resettable CNMC.
- W [DNO18, ADN+19b] pokazaliśmy pierwszą konstrukcję resettable CNMC w 8−split state model w wersji super-silnej[13]. Był to zdecydownie największy projekt, jakim się zajmowałem. W wariancie resettable występuje całkiem nowe zjawisko niespotkane we wcześniejszych problemach: *small incremental learning*. Jeśli przyjrzymy się dowodowi niemożności konstrukcji resettable w modelu 2 split state, to zauważymy, że główna idea była, by przeciwnik uczył się małych kawałków informacji (pojedynczych bitów) i akumulował je tak długo, aż złamał schemat, co więcej robił to bez ryzyka detekcji. To *incremental learning* jest niestety nieuniknione, za każdym razem gdy przeciwnik otrzymuje wiadomość same to uczy się, że słowo kodowe jest w zbiorze punktów stałych manipulacji. Kiedy otrzymuje wiadomość c'[14], to wie, że oryginalne słowo kodowe jest w przeciwobrazie c' w jego funkcji manipulacji. Jest zatem jasne, że by zbudowac resettable CNMC trzeba umieć sobie

[13]Definicja znajduje sie we wcześniejszej dyskusji wyników z [AKO17] w sekcji 4.5. Pobieżnie mówiąc, wariant silny oznacza, że $m' = m$ tylko jeśli słowo kodowe się nie zmieniło, zaś super-silny oznacza, że ponad to jeśli słowo kodowe się zmieniło to nie tylko m' jest niezależne od m ale całe zmanipulowane słowo kodowe c' jest niezależne od m.

[14]W wariancie super-silnym przeciwnik otrzymuje całe zmanipulowane słowo kodowe jeśli tylko jest to słowo różne od oryginalnego i nie generujące błędu dekodera. Słowo c' nie powinno zdradzać żadnej informacji o oryginalnej wiadomości

poradzić z incremental learning. Osiągamy to, wprowadzając serie nowych technik. W szczególności kontrolujemy sposób, w jaki przeciwnik może gromadzić wiedzę, kształt[15] zbioru wszystkich słów kodowych, które są w przeciwobrazie nagromadzonej wiedzy, no i rozkład w obrębie tego zbioru. W końcu pokazujemy, że gdy przeciwnik stara się zdobyć nową wiedzę, to musi zaryzykować detekcję, co więcej prawdopodobieństwo detekcji jest proporcjonalne do ilości wiedzy, którą chce posiąść. O ile to ostatnie zdanie ma sens intuicyjnie to ścisłe uchwycenie zależności między detekcją a ilością nowej wiedzy jest istotnym problemem i wymyka się wszelkim znanym wariantom entropii czy min-entropii (nie jest to kwestia techniczna, po prostu istnieją kontrprzykłady). W związku z tym wymyśliliśmy inny sposób kontroli przepływu informacji poprzez deathzones lemma. Dla przykładu załóżmy, że przeciwnik w danej rundzie manipulacji może uzyskac trzy odpowiedzi: same, c_1 lub c_2, popatrzmy na zbiów słów kodowych, które dają same nazwijmy go A_0, zbiór słów kodowych, które dają c_1 oznaczmy jako A_1, w końcu przeciwobraz c_2 oznaczmy jako A_2. W pracy pokazujemy, że pozostałe słowa kodowe (tj. słowa dające błąd dekodera) mogą zostać podzielone na trzy zbiory D_0, D_1, D_2 takie, że

$$P(A_i|A_i \cup D_i) \approx 2^{-\log \frac{1}{P(A_i)}}$$

Ta zależność oznacza dokładnie to, na czym nam zależało, każdy zbiór A_i jest spokrewniony ze *strefą detekcji* D_i, o rozmiarach proporcjonalnych do ilości informacji, której nauczyłby się przeciwnik gdyby słowo kodowe wypadało w zbiorze A_i (tj. gdyby dostał odpowiedz same dla $i = 0$, lub c_i dla $i = 1, 2$). Co istotne cena detekcji jest obliczana indywidualnie dla każdego ze zbiorów, jest to absolutnie kluczowe, i ilustruje dlaczego żaden standardowy argument entropijny nie ma szans tutaj zadziałać - ta technika nie lubi uśredniania.

- Ostatnia praca z serii CNMC nie jest w modelu split-state. W [DKO+18] budujemy CNMC przeciwko permutacjom i nadpisaniom- przeciwnik może permutować, kopiować, kasować i nadpisywać bity (ale nie może ich odwracać). Model ten wydaje się dość sztuczny, ale ma bardzo ciekawe zastosowanie do schematów zobowiązań.

  Schematy zobowiązań to protokoły, w których Alicja rozmawia z Bobem i zobowiązuje się do dowolnej wybranej przez siebie wartości, wartość ta pozostaje ukryta dla Boba do momentu, kiedy Alicja postanowi "otworzyć" swoje zobowiązanie. Oczywiście, by to miało jakikolwiek sens, Alicja nie jest w stanie zmienić wartości, do której się zobowiązała. Niekowalne schematy zobowiązań to protokoły, w których Alicja rozmawia z Bobem w obecności Man-in-the-Middle(MiM). Każda wiadomość jest czytana przez MiM i może być zmanipulowana. Niekowalność schematu wymaga, by MiM nie był w stanie utworzyć i otworzyć zobowiązania do skorelowanej (ale innej) wiadomości do tej wybranej przez Alicję. W tej pracy rozważamy silniejszy wariant tej definicji zwany parallel CCA.

  Z pomocą naszego CNMC pokazujemy jak zbudować schemat parallel CCA dla ciągu bitów za pomocą schematu parallel CCA dla pojedyńczego bitu. To uzupełnia pewną lukę. Już wcześniej wiedzieliśmy, że schematy zobowiązań dla pojedyńczego bitu implikują schematy zobowiązań dla dłuższych wartości (to oznacza, że budowa schematu dla wielobitowych wartości nie wymaga żadnych dodatkowych założeń obliczeniowych). Teraz wiemy, że to samo jest prawdą dla schematów parallel CCA.

  Ściśle mówiąc, pokazujemy jak nasz CNMC może zostac użyty do rozbudowania schematu zobowiązań self-destruct parallel CCA dla jednobitowych wiadomości do schematu self-destruct parallel CCA dla dowolnych wiadmości, gdzie self-destruct parallel CCA jest słabszą formą parallel CCA security. Następnie pozbywamy się ograniczenia self-destruct z użyciem funkcji jednokierunkowych (które i tak są konieczne do budowy schematów zobowiązań), co daje pełny wynik.

[15] Kontrola kształtu wiedzy jest konieczna by dowodzić niezależności przewnych fragmentów słowa kodowego nawet podczas procesu uczenia, jest to absolutnie kluczowe dla zastosowania lematów ekstraktorowych. Dla przykładu, wyobraźmy sobie zmienne losowe (X, Y) samplowane jednostajnie z kwadratu - wtedy X, Y są jednostajne, jednak, jeśli będziemy je samplować z koła to już niezależne nie będą.

4.8 Niekowalne schematy dzielenia sekretu (Non-malleable secret sharing)

Następny obiekt jest niezwykle interesujący z wielu powodów: w jego kontekscie model split state jest bardzo naturalny, używa 2−split state NMC jako budulca, i można na niego patrzeć jako na uogólnienie pojęcia kodów niekowalnych w modelu split state.

4.8.1 Wstęp do schematów dzielenia sekretu i do schematów threshold secret sharing

Schematy dzielenia sekretu są jednymi z najstarszych i najbardziej podstawowych obiektów w kryptografii. Zostały wprowadzone w latach 70-tych przez Shamira i Blakely'ego i do dziś pozostają podstawowymi składnikami wielu protokołów kryptograficznych.

Wyobraźmy sobie n graczy, chcemy rozdzielić sekret pomiędzy nich w taki sposób, że żadne $n-1$ spośród graczy nie jest w stanie *zrekonstruować* sekretu, ale wszyscy n gracze razem mogą w pełni odtworzyć ów sekret. Najprostszym schematem dzielenia sekretu jest XOR-secret sharing: by rozdzielić sekret/wiadomość m wybieramy jednostajnie losowe $a_1, ..., a_{n-1}$, każdy z graczy jako swój udział (ang. share) otrzymuje a_i, poza ostatnim graczem, ten otrzyma: $a_1 \oplus a_2 \oplus ... \oplus a_{n-1} \oplus m$. Ponieważ wszystkie a_i są jednostajne i niezależne, jeśli "zgubimy" choć jedno z nich, nie będziemy w stanie odtworzyć sekretu. Taki schemat nazywamy n out of n secret sharing. Naturalnym rozszerzeniem jest t out of n secret sharing, gdzie każde t graczy może odtworzyć sekret, ale dowolne $t-1$ już nie. Przykładem jest schemat Shamira: wybieramy losowy wielomian stopnia $t-1$, taki, że $m = p(0)$, gracz i'ty otrzymuje jako swój udział wartość $p(i)$. Jest jasne, że dowolne t wartości wystarcza, by odtworzyć $p(0)$, ale $t-1$ wartości nie daje żadnej informacji o sekrecie.

Możemy rozważać kolejne uogólnienia. Niech $\mathcal{A}$, będzie dowolną, monotoniczną[16] strukturą podzbiorów zbioru $\{1, ..., n\}$. Powiemy, że schemat jest schematem dzielenia sekretu ze strukturą dostępu (ang. access structure) $\mathcal{A}$ jeśli dla każdego zbioru $A \in \mathcal{A}$ gracze ze zbioru A mogą odtworzyć sekret, podczas gdy gracze ze zbioru $B \notin \mathcal{A}$ nie mogą uzyskać żadnej informacji o dzielonym sekrecie. W szczególności dla schematu t out of n struktura dosępu ma formę: $\mathcal{A} = \{A \subseteq \{1, ..., n\} \,|\, |A| \geq t\}$.

4.8.2 Niekowalne schematy dzielenia sekretu (ang. Non-malleable secret sharing)

W [GK18a] autorzy wprowadzili pojęcie niekowalnych schematów threshold secret sharing. Jest to naturalne rozszerzenie kodów niekowalnych w split state i schematów dzielenia sekretu. Dzielimy wiadomość m na udziały $S_1, ..., S_n$, i oczekujemy, że:

1. $(S_1, ..., S_n)$ są schematem t out of n, to znaczy, że t graczy może odtworzyć wiadomość, ale $t-1$ nie.
2. jeśli przeciwnik zmanipuluje $(S_1, ..., S_n) \to (S'_1, ..., S'_n)$ w modelu split state (tj. S'_i zależy tylko od S_i) wtedy po odtworzeniu sekretu uzyskamy albo oryginalny sekret, albo błąd odtworzenia, albo sekret kompletnie niezależny od oryginalnego

Wkrótce po wprowadzeniu pojęcia pojawiła się seria prac [SV18, BS18, KMS18] and [ADN+19a] (wszystkie 4 niezależne), a później pojawiły się [BFV19][BFO+20][KMZ20, CGGL20].

W [GK18a] autorzy koncentrują się na rekonstrukcjach tylko ze sbioru minimalnego. Zbiór $A \in \mathcal{A}$ nazwiemy minimalnym dla struktury dostępu $\mathcal{A}$, jeśli dla każdego podzbioru $A' \subset A$ zachodzi $A' \notin \mathcal{A}$. To ograniczenie ma sens, mając nie-minimalny zbiór graczy A zawsze możemy wybrać minimalny podzbiór $A' \subset A$ (powiedzmy, że to pierwszy minimalny podzbiór w jakimś porządku), a następnie rekonstruować sekret, używając A'. Jednak istnieją scenariusze, w których nie chcemy tak ingerować w procedurę rekonstrukcji, ani po prostu ignorować graczy ze zbioru $A \setminus A'$.

W [ADN+19a] pokazujemy ogranicznenia definicji opierającej się na zbiorze minimalnym. Okazuje się ona istotnie słabsza od ogólnej definicji (gdzie procedura rekonstrukcji nie ogranicza się do zbioru minimalnego). My rozważamy trzy warianty definicji bezpieczeństwa:

[16] Jeśli zbiór $A \in \mathcal{A}$ to każdy zbiór zawierający A też jest w $\mathcal{A}$.

1. Bezpieczeństwo tylko dla rekonstrukcji ze zbioru minimalnego (oryginalna definicja z [GK18a]). Po manipulacji udziałów przeciwnik wybiera minimalny podzbiór A i zrekonstruowana wiadomość jest albo równa oryginalnej, albo kompletnie niezależna (włączając błąd rekonstrukcji).

2. Bezpieczeństwo dla rekonstrukcji z dowolnego zbioru. Tutaj schemat zadziała dla dowolnego $A \in \mathcal{A}$, odtworzona wiadomość jest **zawsze** albo równa oryginalnej, albo od niej niezależna.

3. Bezpieczeństwo przy wielokrotnej manipulacji. Tutaj przeciwnik może manipulować udziałami wielokrotnie, oraz rekonstruować z dowolnego zbiorów (niekoniecznie minimalnego) po każdej manipulacji. Definicja bezpieczeństwa jest taka sama jak w przypadku CNMC- wszystkie odtworzone sekrety razem pozostają niezależne od oryginalnego sekretu[17]. W [BS18] autorzy rozważają podobny model, ale pozwalają tylko na rekonstrukcje z jednego ustalonego zbioru.

W [ADN+19a] uzyskaliśmy następujące wyniki:

- *Separacja*: Istnieją schematy spełniające definicję 1, ale nie spełniające definicji 2 - jest to potencjalnie katastroficzna wada definicji 1. Używając takich schematów, musimy pilnować, by wszystkie operacje (również te w aplikacji) odbywały się tylko na zbiorach minimalnych. Ponadto pokazujemy separacje pomiędzy definicjami 2 i 3.

- *Kompilator:* Podajemy generalny kompilator zamieniający dowolny schemat o 3-monotonicznej strukturze dosępu (tj. strukturze dostępu, w której najmniejszy zbiór ma 3 elementy), w niekowalny schemat o tej samej strukturze dostępu bezpieczny według definicji 3.

- *Wycieki:* Pokazujemy też jak zmodyfikować ten kompilator, by uzyskać schemat odporny na wycieki o bardzo dobrym rate. W tym wariancie przeciwnik nie wykonuje ataku aktywnego. Zamiast tego pasywnie wycieka informacje o udziałach, a nasępnie korumpuje zbiór nie znajdujący się w strukturze dostępu (tak zwany zbiór *niekwaliwikowany* ang. *not qualified set*). Definicja bezpieczeństwa wymaga, że nawet gdy przeciwnik pozna te wycieki oraz wszystkie udziały ze zbioru niekwalifikowanego to wciąż sekret pozostanie dla niego nieznany.

- *Zastosowania:* Pokazujemy zastosowania naszego kompilatora do *non-malleable threshold signatures*. Schematy threshold signature to seria protokołów, które pozwalają dzielić klucz prywatny sk pomiędzy n graczy w taki sposób, że dowolne t graczy może wykonać obliczenie wielopodmiotowe (ang. multi-party computation), które na wyjściu da podpis (przy użyciu klucza prywatnego sk) na danej wiadomości (bez ujawniania sk w dowolnym momencie wykonania protokołu).

  Schemat *non-malleable threshold signature* jest schematem threshold signature scheme o dodatkowej własności: nawet jeśli przeciwnik zmanipuluje lokalne udziały graczy i będzie obserwował procedurę podpisu na wiadomości, którą sam wybrał, to niczego nie dowie się o tajnym kluczu prywatnym sk. Główny pomysł jest dość naturalny: użyjemy niekowalnego schematu dzielenia sekretu, by podzielić sk na udziały. Po manipulacji tajny klucz pozostanie niezmieniony lub zostanie całkowicie zniszczony. Więc przeciwnik nie będzie w stanie wykonać ataku skorelowanego klucza. W przypadku gdy $sk' = sk$ bezpieczeństwo samego schematu podpisu gwarantuje, że przeciwnik nie nauczy się niczego. To wszystko gwarantuje w szczególności, że przeciwnik nie jest w stanie podrobić podpisu, nawet obserwując wykonanie protokolu, którym sam manipulował.

4.8.3 Kompilator: intuicja

Zadanie jest następujące: chcemy zamienić dowolny schemat dzielenia sekretu z dowolną strukturą dostępu na schemat niekowalny o tej samej strukturze dostępu. Konstrukcja jest bardzo prosta:

[17] Jak wcześniej, jeśli $m'_i = m$ to zastępujemy m'_i symbolem same.

1. Mając sekret m, dzielimy go przy użyciu oryginalnego (nie niekowalnego) schematu na udziały $S_1, ..., S_n$
2. Wybieramy[18] jednostajnie losowe $P \in \{0,1\}^p$
3. Używając kodu niekowalnego w 2 split state, kodujemy wiadomości $P||S_i$ w słowa kodowe L_i, R_i
4. Nowy i'ty udział wygląda następująco:

$$(R_1, ..., R_{i-1}, L_i, R_{i+1}, ..., R_n)$$

Niestety dowód bezpieczeństwa tego kompilatora, zwłaszcza w wariancie 3, jest długi i techniczny więc zamiast tego skupimy się na komilatorze dającym bezpieczeństwo przeciwko wyciekom:

1. Mając sekret m, dzielimy go przy użyciu oryginalnego (nie niekowalnego) schematu na udziały $S_1, ..., S_n$
2. Używając dowolnego *seeded extractor*(definicja poniżej) Ext znajdujemy $\mathsf{Seed_i}$ i X_i jednostajnie losowe takie, że $\mathsf{Ext}(X_i, \mathsf{Seed_i}) = S_i$
3. Nowy i'ty udział wygląda nasępująco:

$$(\mathsf{Seed_1}, ..., \mathsf{Seed_{i-1}}, X_i, \mathsf{Seed_{i+1}}, ..., \mathsf{Seed_n})$$

By naszkicować dowód, najpierw musimy opowiedzieć o własnościach głównego budulca: *seeded extractor'a*. Oryginalną motywacją stojącą za tym obiektem była symulacja randomizowanych algorytmów przy użyciu źródeł losowości o wysokiej min-entropii (zamiast źródeł jednostajnie losowych, które ciężko znaleźć w przyrodzie). *Seeded extractor* Ext bierze na wejściu zmienną losową X (zwaną *źródłem* ang. *source*) o pewnej min-entropi, oraz bardzo krótki (znacząco krótszy niż wyjście ekstraktora) jednostajny wektor Seed (zwane seed). Na wyjściu ekstraktor zwraca jednostajnie losowy ciąg bitów, wyjście pozostaje jednostajne nawet jeśli ujawnimy seed (więc możemy wielokrotnie używać tego samego seed, tak długo jak źródła pozostają od niego niezależne).

Sposób w jaki używamy seeded extractors w kryptografii ma mało wspólnego z oryginalnym zastosowaniem. Możemy ich używać do ochrony przed wyciekami. Dzięki pewnym trikom technicznym możemy pokazać, że wyjście ekstraktora pozostaje ukryte/jednostajne nawet jeśli przeciwnik pozna większość informacji o źródle X poprzez wyciek, i do tego dostanie dosęp do całego Seed. Należy tutaj poczynić istotną uwagę, że aby mieć jakikolwiek pożytek z tego obiektu musimy być w stanie wydajnie wykonać krok 2 kompilatora. Ta własność zwana jest *wydajnym samplowaniem z przeciwobrazu* (ang. *efficient preimage sampling*) i nie wszystkie seeded extractors ją mają.

Szkic dowodu: Po pierwsze zauważmy, że i'ty gracz jest jedynym trzymającym X_i więc udział S_i (pochodzący z oryginalnego schematu) pozostaje ukryty chyba, że i'ty gracz uczestniczy w rekonstrukcji. Ponadto zauważmy, że każdy inny gracz zna $\mathsf{Seed_i}$, więc gracz i przy pomocy każdego innego gracza jest w stanie odtworzyć oryginalny udział S_i. To oznacza, że nowy schemat ma tę samą strukturę dostępu co oryginalny schemat (ten bez odporności na wycieki).

Pozostaje jedynie uargumentować, że nowy schemat jest odporny na wycieki. Dla jasności obrazu skupimy się tylko na wyciekach i przeciwnik nie będzie miał dostępu do zbioru niekwalifikowanego. Przeciwnik

[18] Ten krok jest konieczny by zagwarantować spójność przy rekonstrukcjach ze zbiorów nieminimalnych.

wybiera $f_1, ..., f_n$ dowolne funkcje o *małym*[19] wyjściu. Chcemy pokazać, że:

$$f_1((X_1, \mathsf{Seed}_2, ..., \mathsf{Seed}_\mathsf{n})),$$
$$f_2((\mathsf{Seed}_1, X_2, \mathsf{Seed}_3, ..., \mathsf{Seed}_\mathsf{n})),$$
$$...$$
$$f_n((\mathsf{Seed}_1, ..., \mathsf{Seed}_{\mathsf{n}-1}, X_n)),$$

wszystkie razem nie ujawniają żadnej informacji o ukrytym sekrecie. Trik techniczny polega na tym, żeby ujawnić wszystkie $\mathsf{Seed}_1, ..., \mathsf{Seed}_\mathsf{n}$. Zauważmy, że to zmnienia wyciek $f_i((\mathsf{Seed}_1, ..., \mathsf{Seed}_{\mathsf{i}-1}, X_i, \mathsf{Seed}_{\mathsf{i}+1}, ..., \mathsf{Seed}_\mathsf{n}))$ w zwykły wyciek z X_i, tj. w jakieś $\tilde{f}_i(X_i)$. Należy tutaj podkreślić, że ten wyciek nie zależy od Seed_i. Jak już mówiliśmy seeded extraktor chroni swoje wyjście (czyli S_i) przed wyciekiem nawet jeśli ujawnimy cały seed i większość informacji o źródle. W związku z tym przeciwnik nie poznał żadnego S_i a co za tym idzie oryginalny sekret pozostał bezpieczny.

Dowód niekowalności nie podąża tą samą scieżką. Zamiast tego wykorzystuje serię redukcji hybrydowych i dowodzi, że jeśli umielibyśmy złamać bezpieczeństwo kompilatora, to używając tej samej strategii, moglibyśmy złamać bezpieczeństwo kodu niekowalnego użytego do budowy kompilatora.

5 Inne serie tematyczne, niezwiązane z głównym cyklem

5.1 Estymatory entropii Rényi'ego

Entropia Rényi'ego jest uogólnieniem entropii Shannona, ma wiele zastosowań do uczenia maszynowego, przetwarzania obrazu, detekcji anomalii w sieci, analizy przepływu informacji i wiele innych. Jest sparametryzowana wartością $\alpha \geq 0$. Dla zmiennej losowej X definiujemy ją następująco:

$$\mathbf{H}_\alpha(X) = \frac{1}{1-\alpha} \log \left(\sum_{i=1}^{n} P(X=i)^\alpha \right).$$

Jeśli $\alpha \to 1$ wtedy entropia Rényi'ego zbiega do entropii Shannona, jeśli $\alpha \to \infty$ wtedy entropia Rényi'ego zbiega do min-entropii. Przypadek $\alpha = 2$ jest kolejną znaną entropią zwaną *entropią kolizyjną* (ang. *collision entropy*).

Mamy serię dwóch prac rozważających problem wydajnej estymacji entropii:

Question 5.1. *Mając dostęp do sampli ze źródła losowości X o nośniku mającym K elementów, pytamy ile porzeba próbek z X by estymować entropie Rényi'ego źródła X?*

Są dwa główne warianty tego pytania. W jednym zakładamy, że próbki/sample z X są i.i.d., w drugim zakładamy, że X jest źródłem Markova (tj. próbki tworzą łańcuch Markova).

- W [OS17] poprawiamy zarówno dolne jak i górne ograniczenia z pracy [AOST15]. Prowadzimy dokładniejszą analizę ich estymatora i pokazujemy jego poprawioną złożoność (ilość sampli, które potrzebuje). Ponadto rozważamy inny ciekawy wariant głównego problemu- co jeśli wiemy, że źródło ma entropię ograniczoną z góry (ważne dla detekcji anomalii), przewaga estymatora w tym wariancie jest gigantyczna (i zaskakująca) zwłaszcza w reżimach niskiej entropii. Używając metody Le Cam'a, pokazujemy również, że nasz estymator jest optymalny.
  - Górne ograniczenie w przypadku ograniczonej entropii: wymagamy $n = O(1) \cdot 2^{\left(1-\frac{1}{\alpha}\right)H_\alpha}$ sampli dla całkowitych wartości $\alpha > 1$, jest to analiza worst-case dla wszystkich rozkładów o entropii Renyi'ego równiej lub mniejszej niż H_α.

[19] Tylko odrobinę mniejszym niż długość całego udziału.

  - Górne ograniczenie bez gwarancji na entropię źródła: wymagamy $n = O(1) \cdot K^{1-\frac{1}{\alpha}}$ sampli, jest to worst case dla wszystkich distribucji o nośniku K elementowym.
  - Dolne ograniczenie: $n = \Omega(1) \cdot K^{1-\frac{1}{\alpha}}$ sampli dla dowolnego rzeczywistego $\alpha > 1$, jest to worst case dla wszystkich distribucji o nośniku K elementowym. Stała jest proporcjonalna do odwróconego wielomianu od dokładności.

- W [OS20] rozważamy problem estymacji entropii Rényi'ego dla źródeł Markova. W [KV16] autorzy pokazali, że dobry mixing time łańcucha Markova jest konieczny, by estymować entropię. My pokazujemy, że nawet przy bardzo dobrych własnościach mixingu, estymacja entropi dla $\alpha > 1$ wymaga przynajmniej $\Omega(K^{2-1/\alpha})$ sampli, gdzie K jest rozmiarem nośnika/alfabetu. W szczególności min-entropia wymaga $\Omega(K^2)$ sampli, a entropia kolizyjna przynajmniej $\Omega(K^{3/2})$ sampli.

  Pod pewnymi naturalnymi założeniami (jak na przykład przyzwoity mixing time, który tak czy owak jest konieczny do tego zadania) możemy pokazać estymator wymagający $n = O(|S|^2) \log^{O(1)}(|S|)$ sampli. To pasuje do naszego dolnego ograniczenia dla min-entropii. Otrzymaliśmy te wyniki konstruując serię macierzy o odpowiednich własnościach spektralnych. Używając teorii perturbacji, pokazujemy, że własności spektralne tych macierzy są bardzo wrażliwe na najmniejsze zakłócenia, to pozwala nam użyć metody Le Cam'a i osiągnąć dolne ograniczenia.

5.2 Losowość i problem ekstrakcji

Pojęcie losowości odgrywa centralną rolę w informatyce teoretycznej. Wiele problemów algorytmicznych jest istotnie prostszych do rozwiązania jeśli algorytm ma dostęp do losowego ciągu bitów X. Również w kryptografii większość obiektów, schematów i protokołów jak na przykład kryptografia klucza publicznego jest niemożliwa bez dostępu do losowości. Zwykle zakładamy w tych protokołach, że mamy dostęp do jednostajnej losowości. Jednak jednostajnie losowe bity nie są tak łatwe do zdobycia. Jak już wspominaliśmy wcześniej, możemy użyć *seeded extractors* by przy użyciu niewielkiego ciągu jednostajnego wyciągać/ekstrachować losowość z rozkładów o wysokiej min-entropii. Alternatywą jest użycie dwóch lub więcej niezależnych źródeł o pewnej min-entropi (ten proces nie wymaga jednostajnie losowego seed).

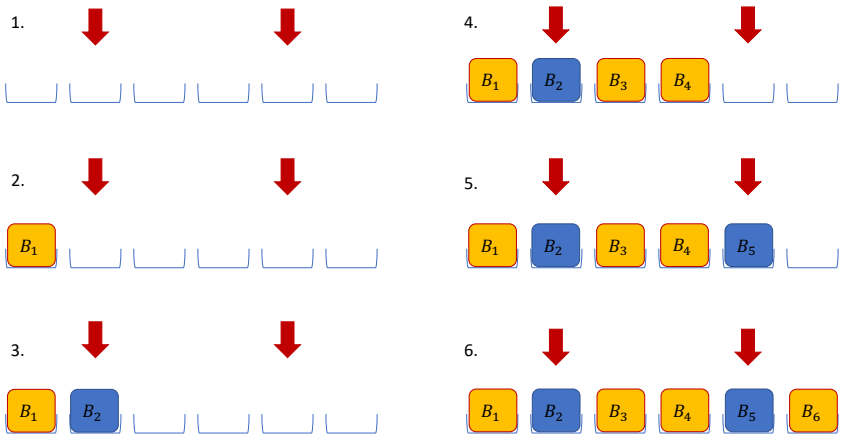
- W [AOR+20] zajmujemy się probleme ekstrakcji ze źródła typu *SHELA*. Źródło SHELA (rys. 2) modeluje nasępujący scenariusz: Istnieje wiele źródeł losowości, niektóre produkują i publikują losowe ciągi bitów o pewnej min-entropii, inne źródła są skorumpowane i produkują ciągi bitów o dowolnej korelacji z wszystkim co zostało opublikowane w przeszłości (my nie wiemy, które są które).

  Pokazujemy, że w takim scenariuszu nie jest możliwe uzyskanie jednostajnie losowych bitów. Jednak wciąż można uzyskać coś niemal równie przydatnego: źródło *somewhere-uniform*. Źródło somewhere uniform składa się z wielu bloków i gwarantuje, że przynajmniej pare z tych bloków jest jednostajnych i niezależnych od siebie. Pokazujemy różne konwersje parametrów (ang. trade-offs). Ponadto podajemy przykłady zastosowań źródeł somewhere uniform do symulacji randomizowanych algorytmów i do non-interactive witness indistinguishable proofs.

- W [AGO+20] częściowo inspirowani [AOR+20], przyglądamy się ponownie fundamentalnemu problemowi dolnych ograniczeń dla seeded ekstraktorów i ich różnych, naturalnych wariantów. Te warianty biorą się z zamiany kwantyfikatorów dla seedów. Silne seeded ekstraktory wymagają, żeby średnie odchylenie od jednostajności (średnia brana nad wszystkimi seedami) było małe dla wszystkich rozkładów źródeł o danej min-entropii. Somewhere extractors wymagają jedynie, żeby dla każdego źródła istniał seed, dla którego odchylenie od jednostajności jest małe. Uogólniając, możemy patrzeć na obiekty, które nazwaliśmy probable extractors, dla których dla każdego rozkładu źródła istniej duży ułamek seedów generujących małe odchylenie od jednostajności. Takie obiekty odgrywają istotną rolę w wielu konstrukcjach obiektów pseudolosowych, chociaż nie były wcześniej analizowane i zdefiniowane wprost.

Wcześniejsze techniki uzyskiwania dolnych ograniczeń nie dają satysfakcjonujących rezultatów dla obiektów zdefiniowanych powyżej. Wymyśliliśmy więc nowe podejście dające istotnie poprawione ograniczenia dolne dla somewhere extractors i dla probable extractors. Te dolne ograniczenia uzupełniamy konstrukcją somewhere extraktora o pasujących parametrach, co oznacza, że nasze ograniczenia są optymalne (przynajmniej w reżimie wysokich entropii). Jest to bardzo zaskakujące gdyż oznacza to, że losowe funkcje są dalekie od osiągania optymalnych parametrów dla tego typu obiektów[20]. Nasze nowe techniki dają również alternatywny i dużo prostszy dowód (z jawnymi stałymi) bardzo ważnego wyniku ograniczeń dolnych na długość jednostajnego seed w seeded extractors oryginalnie osiągniętego przez Radhakrishnan i Ta-Shma (SIAM J. Discrete Math., 2000).

Należy tutaj wspomnieć, że źródła somewhere uniform i somewhere extractors są silnie połączone. Jeśli Ext jest somewhere extractorem, oraz X ma pewną min-entropię, to jeśli przeiterujemy po wszystkich seedach: $s_1, ..., s_D$ wtedy $(\mathsf{Ext}(X, s_1), ..., \mathsf{Ext}(X, s_D))$ jest źródłem somewhere-uniform. Strategia iterowania po wszystkich możliwych seedach jest użyteczna w wielu scenariuszach, więc ograniczenia dolne na rozmiar seed tłumaczą się na ograniczenia dolne na rozmiar źródła, co z kolei tłumaczy się na wydajność protokołów używających takich źródeł.



Rysunek 2: Procedura samplowania/próbkowania ze źródła SHELA. 1) Przeciwnik wybiera które pozycje będą uczciwie losowe. 2) Przeciwnik produkuje pierwszy blok. 3) Uczciwie losowy blok jest losowany niezależnie od B_1. 4) Przeciwnik produkuje kolejne bloki. W przeciwieństwie do bloku B_1, bloki B_3, B_4 mogą zależeć od uczciwego bloku B_2. 5) Uczciwy blok jest losowany niezależnie od bloków B_1, B_2, B_3, B_4. 6) Przeciwnik produkuje ostatni blok, który może zależeć od wszystkich poprzednich.

[20]Losowe funkcje mają z prawdopodobieństwem bliskim 1 parametry optymalne dla seeded extractors, kodów niekowalnych, niekowalnych ekstraktorów etc. Zwykle problemem jest istnienie wydajnych konstrukcji, wszak losowa funkcja nie jest wydajna. To podkreśla wyjątkowość obiektów typu somewhere i probable na tle innych obiektów pseudolosowych

6 Informacja o wykazywaniu się istotną aktywnością naukową albo artystyczną realizowaną w więcej niż jednej uczelni, instytucji naukowej lub instytucji kultury, w szczególności zagranicznej.

Z wyjątkiem [DKO13] (opublikowana przed moim doktoratem) i [AO20], wszystkie moje prace (nie tylko te uwzględnione w głównym cyklu zostały) napisane przy współpracy z innymi uniwersytetami, w każdym przypadku przynajmniej jeden współautor był z zagranicznej jednostki naukowej.

W przypadku [AO20] obaj byliśmy zatrudnieni na jednym uniwersytecie - National University of Singapore.

6.1 Słowo o cyklu stanowiącym podstawę tej habilitacji (t.j. osiągnięcie, o którym mowa w art.219...)

Pierwsze dwie prace cyklu ([ADKO15b, ADKO15a]) zostały napisane podczas mojego zatrudnienia na Uniwersytecie Warszawskim. Pierwsza powstała przy współpracy z New York University(NYU), druga z École Polytechnique Fédérale de Lausanne(EPFL) w Szwajcarii

Następne trzy prace ([AKO17, DKO+18, ADN+19b]) zostały napisane i opublikowane podczas mojego zatrudnienia na Uniwersytecie w Aarhus (Dania). Projekt [AKO17] został rozpoczęty w Warszawie, kontynuowany w Aarhus i zakończony podczas wizyt na EPFL w Szwajcarii. Pierwsza praca powstała przy współpracy z EPFL, druga przy współpracy z Warszawą, Salerno, Aarhus i Singapurem, trzecia z Aarhus, Friedrich-Alexander-University Erlangen-Nuremberg i Singapurem.

Ostatnie dwie prace cyklu ([ADN+19a, AO20]) zostały napisane i opublikowane podczas mojego zatrudnienia na National University of Singapore, pierwsza przy współpracy z Aarhus i Salerno.

7 Dane wymagane przez Uniwersytet Warszawski

7.1 Wizyty, odczyty, konferencje, zaproszone odczyty(invited speaker) etc.

7.1.1 Wystąpienia na Konferencjach

Przed Doktoratem:

- CRYPTO (Santa Barbara, CA, USA): "NMC from 2-Source Extractors", 2013

Po Doktoracie:

- TCC (Baltimore, MD, USA): "Inception Makes NMC Stronger", 2017
- ISIT (Vail, CO, USA): "Inverted Leftover Hash Lemma", 2018
- ISIT (Vail, CO, USA): "Optimal Leakage Resilient AMD", 2018
- TCC (Goa, Indie): "Continuous NMC Against Permutations and Overwrites", 2018
- EUROCRYPT (Darmstadt, Niemcy): "Continuous NMC in 8-Split State", 2019

7.1.2 Wizyty Naukowe i Zaproszone Odczyty

Przed Doktoratem:

- **Odczyt** Workshop on Leakage, Tampering and Viruses, Warszawa (Polska), "NMC from 2-Source Extractors", 2013

Po Doktoracie:

- **Odczyt** Forum for Theoretical Informatics, Warszawa (Polska): "Non-Malleable Reductions", 2015
- **Wizyta i Odczyt** EPFL (Szwajcaria): "Inception Makes NMC Stronger", 2015
- **Wizyta** EPFL (Szwajcaria), 2015
- **Wizyta i Odczyt** IDC Herzelia (Izrael): "Inverted Leftover Hash Lemma", 2017
- **Wizyta i Odczyt** ETH (Szwajcaria): "Continuous NMC in Constant Split-State", 2018
- **Wizyta** NUS (Singapur), twice, 2018
- **Wizyta i Odczyt** IST Austria (Austria): "Continuous NMC in 8-Split State", 2018
- **Wizyta** University of Salerno (Włochy), 2019
- **Zaproszony Odczyt (Invited Speaker)**, Workshop on Modern Trends in Cryptography, Nanyang Technological University: "Introduction to Continuous Non-Malleable Codes in Split-State Model", 2019
- **Wizyta i Odczyt** Aarhus University (Denmark): Extractors Lower Bounds Revisited, 2019

7.2 Informacja o recenzowanych pracach naukowych

Wielokrotnie i regularnie recenzowałem prace dla prestiżowych konferencji jak: *STOC, CRYPTO, EUROCRYPT, TCC, AsiaCRYPT, PKC*, oraz dla wielu czasopism międzynarodowych włączając wysoce ceniony *Journal of Cryptology.*

Literatura

[ADKO15a] Divesh Aggarwal, Yevgeniy Dodis, Tomasz Kazana, and Maciej Obremski. Non-malleable reductions and applications. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 459–468, 2015.

[ADKO15b] Divesh Aggarwal, Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Leakage-resilient non-malleable codes. Theory of Cryptography (TCC), 2015.

[ADL14] Divesh Aggarwal, Yevgeniy Dodis, and Shachar Lovett. Non-malleable codes from additive combinatorics. In *STOC*. ACM, 2014.

[ADN+19a] Divesh Aggarwal, Ivan Damgård, Jesper Buus Nielsen, Maciej Obremski, Erick Purwanto, Joao Ribeiro, and Mark Simkin. Stronger leakage-resilient and non-malleable secret-sharing schemes for general access structures. *CRYPTO*, 2019.

[ADN+19b] Divesh Aggarwal, Nico Dottling, Jesper Buus Nielsen, Maciej Obremski, and Erick Purwanto. Continuous non-malleable codes in the 8-split-state model. Eurocrypt, 2019.

[AGM+14] Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. Explicit non-malleable codes resistant to permutations and perturbations. *IACR Cryptology ePrint Archive*, 2014:841, 2014.

[AGM+15] Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. A rate-optimizing compiler for non-malleable codes against bit-wise tampering and permutations. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, pages 375–397, 2015.

[AGO+20] Divesh Aggarwal, Siyao Guo, Maciej Obremski, João Ribeiro, and Noah Stephens-Davidowitz. Extractor lower bounds, revisited. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[AGV09] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In Omer Reingold, editor, *Sixth Theory of Cryptography Conference — TCC 2007*, volume 5444 of *Lecture Notes in Computer Science*. Springer-Verlag, 2009.

[AK96] Ross Anderson and Markus Kuhn. Tamper resistance — a cautionary note. In *The Second USENIX Workshop on Electronic Commerce*, pages 1–11, November 1996.

[AKO17] Divesh Aggarwal, Tomasz Kazana, and Maciej Obremski. Inception makes non-malleable codes stronger. In *Theory of Cryptography Conference (TCC)*, pages 319–343. Springer, 2017.

[AKO18] Divesh Aggarwal, Tomasz Kazana, and Maciej Obremski. Leakage-resilient algebraic manipulation detection codes with optimal parameters. ISIT, 2018.

[AO20] Divesh Aggarwal and Maciej Obremski. A constant-rate non-malleable code in the split-state model. FOCS, 2020.

[AOR+20] Divesh Aggarwal, Maciej Obremski, Joao Ribeiro, Luisa Siniscalchi, and Ivan Visconti. How to extract useful randomness from unreliable sources. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques EUROCRYPT*, pages 343–372. Springer, 2020.

[AOST15] Jayadev Acharya, Alon Orlitsky, Ananda Theertha Suresh, and Himanshu Tyagi. The complexity of estimating rényi entropy. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1855–1869, 2015.

[AS13] Hadi Ahmadi and Reihaneh Safavi-Naini. Detection of algebraic manipulation in the presence of leakage. *Information Theoretic Security - 7th International Conference, ICITS 2013, Singapore, November 28-30, 2013, Proceedings*, pages 238–258, 2013.

[BCL+20] Marshall Ball, Eshan Chattopadhyay, Jyun-Jie Liao, Tal Malkin, and Li-Yang Tan. Non-malleability against polynomial tampering. CRYPTO, 2020.

[BDSG+18] Marshall Ball, Dana Dachman-Soled, Siyao Guo, Tal Malkin, and Li-Yang Tan. Non-malleable codes for small-depth circuits. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 826–837. IEEE, 2018.

[BDSK+19] Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, Huijia Lin, and Tal Malkin. Non-malleable codes against bounded polynomial time tampering. EUROCRYPT, 2019.

[BDSKM16] Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes for bounded depth, bounded fan-in circuits. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 881–908. Springer, 2016.

[BDSKM18] Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes from average-case hardness: Decision trees, and streaming space-bounded tampering. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 618–650. Springer, 2018.

[BFO+20] Gianluca Brian, Antonio Faonio, Maciej Obremski, Mark Simkin, and Daniele Venturi. Non-malleable secret sharing against bounded joint-tampering attacks in the plain model. CRYPTO, 2020.

[BFV19] Gianluca Brian, Antonio Faonio, and Daniele Venturi. Continuously non-malleable secret sharing for general access structures. In *Theory of Cryptography Conference*, pages 211–232. Springer, 2019.

[BGW19] Marshall Ball, Siyao Guo, and Daniel Wichs. Non-malleable codes for decision trees. CRYPTO, 2019.

[Bih94] Eli Biham. New types of cryptanalytic attacks using related keys. *Journal of Cryptology*, 7(4):229–246, 1994.

[BK03] Mihir Bellare and Tadayoshi Kohno. A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In Eli Biham, editor, *Advances in Cryptology—EUROCRYPT 2003*, volume 2656 of *LNCS*. Springer-Verlag, 2003. Full version available at http://www-cse.ucsd.edu/users/tkohno/papers/RKA/.

[BKKV10] Zvika Brakerski, Jonathan Katz, Yael Kalai, and Vinod Vaikuntanathan. Overcomeing the hole in the bucket: Public-key cryptography against resilient to continual memory leakage. In *FOCS*, pages 501–510, Las Vegas, NV, USA, October 23–26 2010. IEEE.

[BS18] Saikrishna Badrinarayanan and Akshayaram Srinivasan. Revisiting non-malleable secret sharing. *IACR Cryptology ePrint Archive*, 2018:1144, 2018.

[CCFP11] Hervé Chabanne, Gérard Cohen, J Flori, and Alain Patey. Non-malleable codes from the wire-tap channel. In *Information Theory Workshop (ITW), 2011 IEEE*, pages 55–59. IEEE, 2011.

[CCP12] Herve Chabanne, Gerard Cohen, and Alain Patey. Secure network coding and non-malleable codes: Protection against linear tampering. In *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*, pages 2546–2550. IEEE, 2012.

[CDF+08] Ronald Cramer, Yevgeniy Dodis, Serge Fehr, Carles Padro, and Daniel Wichs. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In *EUROCRYPT 2008*, April 2008. To Appear.

[CG14] Mahdi Cheraghchi and Venkatesan Guruswami. Capacity of non-malleable codes. In *ITCS*, 2014.

[CGGL20] Eshan Chattopadhyay, Jesse Goodman, Vipul Goyal, and Xin Li. Leakage-resilient extractors and secret-sharing against bounded collusion protocols. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 27, page 60, 2020.

[CGL16] Eshan Chattopadhyay, Vipul Goyal, and Xin Li. Non-malleable extractors and codes, with their many tampered extensions. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 285–298. ACM, 2016.

[CKM11] Seung Geol Choi, Aggelos Kiayias, and Tal Malkin. Bitr: built-in tamper resilience. In *Advances in Cryptology–ASIACRYPT 2011*, pages 740–758. Springer, 2011.

[CKOS19] Eshan Chattopadhyay, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Privacy amplification from non-malleable codes. In Feng Hao, Sushmita Ruj, and Sourav Sen Gupta, editors, *Progress in Cryptology - INDOCRYPT 2019 - 20th International*

Conference on Cryptology in India, Hyderabad, India, December 15-18, 2019, Proceedings, volume 11898 of *Lecture Notes in Computer Science*, pages 318–337. Springer, 2019.

[CZ14] Eshan Chattopadhyay and David Zuckerman. Non-malleable codes in the constant split-state model. *To appear in FOCS*, 2014.

[DDN00] D. Dolev, C. Dwork, and M. Naor. Nonmalleable cryptography. *SIAM*, 30:391–437, 2000.

[DHLAW10] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. Efficient public-key cryptography in the presence of key leakage. In Masayuki Abe, editor, *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 613–631. Springer, 2010.

[DKO13] Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Non-malleable codes from two-source extractors. In *Advances in Cryptology-CRYPTO 2013*. Springer, 2013.

[DKO+18] Ivan Damgård, Tomasz Kazana, Maciej Obremski, Varun Raj, and Luisa Siniscalchi. Continuous nmc secure against permutations and overwrites, with applications to cca secure commitments. TCC, 2018.

[DNO18] Nico Dottling, Jesper Buus Nielsen, and Maciej Obremski. Continuous non-malleable codes in the constant split-state model. unpublished manuscript, 2018.

[DP08] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *49th Symposium on Foundations of Computer Science*, pages 293–302, Philadelphia, PA, USA, October 25–28 2008. IEEE Computer Society.

[DPW10] Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In *ICS*, pages 434–452. Tsinghua University Press, 2010.

[DSK12] Dana Dachman-Soled and Yael Tauman Kalai. Securing circuits against constant-rate tampering. In *Advances in Cryptology–CRYPTO 2012*, pages 533–551. Springer, 2012.

[ECR] ECRYPT. European network of excellence. side channel cryptoanalysis lounge. http://www.emsec.rub.de/research/projects/sclounge.

[FHMV17] Sebastian Faust, Kristina Hostáková, Pratyay Mukherjee, and Daniele Venturi. Non-malleable codes for space-bounded tampering. In *Annual International Cryptology Conference*, pages 95–126. Springer, 2017.

[FMNV14] S. Faust, P. Mukherjee, J. Nielsen, and D. Venturi. Continuous non-malleable codes. In *Theory of Cryptography Conference - TCC*. Springer, 2014.

[FMVW14] S. Faust, P. Mukherjee, D. Venturi, and D. Wichs. Efficient non-malleable codes and key-derivation for poly-size tampering circuits. In *Eurocrypt*. Springer, 2014. To appear.

[FPV11] Sebastian Faust, Krzysztof Pietrzak, and D. Venturi. Tamper-proof circuits: How to trade leakage for tamper-resilience. In *ICALP*. 2011.

[GK18a] Vipul Goyal and Ashutosh Kumar. Non-malleable secret sharing. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 685–698. ACM, 2018.

[GK18b] Vipul Goyal and Ashutosh Kumar. Non-malleable secret sharing for general access structures. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 501–530. Springer, 2018.

[GLM+03] Rosario Gennaro, Anna Lysyanskaya, Tal Malkin, Silvio Micali, and Tal Rabin. Algorithmic Tamper-Proof (ATP) security: Theoretical foundations for security against hardware tampering. In Naor [Nao04], pages 258–277.

[GMW18] Divya Gupta, Hemanta K Maji, and Mingyuan Wang. Constant-rate non-malleable codes in the split-state model. Technical report, Technical Report Report 2017/1048, Cryptology ePrint Archive, 2018.

[GPR16] Vipul Goyal, Omkant Pandey, and Silas Richelson. Textbook non-malleable commitments. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 1128–1141. ACM, 2016.

[GR12] Shafi Goldwasser and Guy N Rothblum. How to compute in the presence of leakage. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 31–40. IEEE, 2012.

[IPSW06] Yuval Ishai, Manoj Prabhakaran, Amit Sahai, and David Wagner. Private circuits II: Keeping secrets in tamperable circuits. In Serge Vaudenay, editor, *Advances in Cryptology—EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 308–327. Springer-Verlag, 2006.

[ISW03] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In Dan Boneh, editor, *Advances in Cryptology—CRYPTO 2003*, volume 2729 of *LNCS*. Springer-Verlag, 2003.

[JW15] Zahra Jafargholi and Daniel Wichs. Tamper detection and continuous non-malleable codes. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, volume 9014 of *Lecture Notes in Computer Science*, pages 451–480. Springer, 2015.

[KMS18] Ashutosh Kumar, Raghu Meka, and Amit Sahai. Leakage-resilient secret sharing. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 25, page 200, 2018.

[KMZ20] Ashutosh Kumar, Raghu Meka, and David Zuckerman. Bounded collusion protocols, cylinder-intersection extractors and leakage-resilient secret sharing. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 27, page 55, 2020.

[KOS17] Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Four-state non-malleable codes with explicit constant rate. In *Theory of Cryptography Conference*, pages 344–375. Springer, 2017.

[KOS18] Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Non-malleable randomness encoders and their applications. In *EUROCRYPT*, pages 589–617. Springer, 2018.

[KV16] S. Kamath and S. Verdú. Estimation of entropy rate and rényi entropy rate for markov chains. In *ISIT*, 2016.

[Li17] Xin Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1144–1156. ACM, 2017.

[Li19] Xin Li. Non-malleable extractors and non-malleable codes: Partially optimal constructions. *CCC'19*, 2019.

[LSW16] Fuchun Lin, Reihaneh Safavi-Naini, and Pengwei Wang. Detecting algebraic manipulation in leaky storage systems. *Information Theoretic Security - 9th International Conference, ICITS 2016, Tacoma, WA, USA, August 9-12, 2016, Revised Selected Papers*, pages 129–150, 2016.

[MR04] Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In Naor [Nao04], pages 278–296.

[Nao04] Moni Naor, editor. *First Theory of Cryptography Conference — TCC 2004*, volume 2951 of *LNCS*. Springer-Verlag, February 19–21 2004.

[NS09] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, volume 5677 of *LNCS*, pages 18–35. Springer-Verlag, 2009.

[OS17] Maciej Obremski and Maciej Skorski. Renyi entropy estimation revisited. APPROX, 2017.

[OS20] Maciej Obremski and Maciej Skorski. Complexity of estimating renyi entropy of markov chains. ISIT, 2020.

[SV18] Akshayaram Srinivasan and Prashant Nalini Vasudevan. Leakage resilient secret sharing and applications. *IACR Cryptology ePrint Archive*, 2018:1154, 2018.

[Wee12] H. Wee. Public key encryption against related key attacks. In *PKC*. 2012.